ЎЗБЕКИСТОН АЛОҚА ВА АХБОРОТЛАШТИРИШ АГЕНТЛИГИ
ТОШКЕНТ АХБОРОТ ТЕХНОЛОГИЯЛАРИ УНИВЕРСИТЕТИ
УРГАНЧ ФИЛИАЛИ

“ТИЗИМЛИ ДАСТУРИЙ ТАЪМИНОТ” ФАНИ БЎЙИЧА МАЪРУЗА МАТНЛАРИ

5521900-“Информатика ва ахборот технологиялари” йўналиши

Урганч - 2009

Тузувчилар: Тошкент ахборот технологиялари университети Урганч филиали “Ахборот технологияларининг” кафедраси асс. Хўжаев О.Қ.

Ушбу қўлланма “Тизимли дастурий таъминот” фани маъруза машғулотларни ўтиш учун асосий қўлланма ҳисобланиб, назарий маълумотлардан ташкил топган

“Ахборот технологиялари” кафедраси

Маъруза № 1.

Мавзу:ЭХМ тизимли дастурий таъминотининг структура ва функциялари.

Режа:

1. **Тизимли дастурий таъминотнинг асосий тушунчалари.**
2. **Дастурлаш тизимлари**
3. **Тизимли дастурий таъминотнинг таркиби ва асосий функциялари.**

1.Тизимли дастурий таъминотнинг асосий тушунчалари

Калит сузлар.

- **Ҳисоблаш тизими**
- **Аппарат кисм**
- **Дастурий кисм**
- **Амалий дастурий таъминот**
- **Тизимли дастурий таъминот**
- **Умумий тизимли дастурий таъминот**
- **Махсус тизимли дастурий таъминот**

Замонавий амалиётда хисоблашларни бажариш учун хисоблаш машиналари ва тизимларидан фойдаланилади.

ЭХМ – бу аппаратуралар мажмуаси булиб, у кандайдир алгоритмлар тупламини харакатларини бажаришни таъминлайди.

Хисоблаш тизими – бу бир ёки бир неча ЭХМ ва дастурлар туплами булиб, узига юклатилган функцияларни бажарилишини таъминлайди.

Шундай килиб, хисоблаш тизимини 2 (икки) булакка булинади:

- дастурий;
- аппарат.

Аппарат кисми уз ичига хисоблаш машинасининг курилмаларининг бача функцияларини киритади. Аппарат кисмининг белгиланиши: киритиш ва чикариш амалларини бажариш, саклаш, узатиш ва маълумот алмашувни амалга оширишдан иборат.

Дастурий кисми эса – бу дастурлар туплами булиб, хисоблаш тизимига юклатилган функцияларни бажарилишини тартибини белгилайди. Дастур натижа олиш максадида аппаратура ишини бошкаради.

Дастурий булак, тизимли дастурий таъминот (ДТ) деб аталувчи бири бирига боглик купгина компоненталар тупламидан ташкил топган.

Изох: «таъминот» сузи хисоблаш тизимига юклатилган функцияларни амалга оширилишига курсатмадир.

Умуман олганда дастурий таъминот тизимини иккита катта булакка булинади:

1. Амалий ДТ (фойдаланувчи учун);
2. Тизимли ДТ.

Амалий дастурий таъминот – бу фойдалунувчиларнинг узлари учун узлари томонидан яратиладиган дастурлардир.

Тизимли дастурий таъминот – бу барча учун яратилган ва универсал булган дастурлардир. У хам икки булакка булинади.:

1. Умумий тизимли дастурий таъминот;
2. Махсус тизимли дастурий таъминот.

Махсус тизимли дастурий таъминот хисоблаш тизимининг аник специфик масалаларини ечиш учун умумий дастурий таъминотга кушилади (учишни бошкариш, харбий масалалар ва х.к.).

Умумий тизимли дастурий таъминот универсал булиб кенг омма вий масалаларни ечиш учун мулжалланган.

Бундан сунг умумий тизимли дастурий таъминотни куриб утамиз. У куйидаги таркибдан иборат:

1. тизимли кайта ишловчи дастурлар;
2. тизимли бошкарувчи дастурлар;
3. кушимча 1 ва 2 каби дастурлар;
4. текширувчи –диагностик дастурлар;
5. амалий дастурлар пакети;
6. Тизимли дастурий таъминот хужжатлари мажмуаси.

1. Тизимли кайта ишловчи дастурлар фойдаланувчиларга хизмат курсатиш масалаларини уларнинг талабларига кура ечишга мулжалланган.

2. Тизимли бошкарувчи дастурлар хисоблаш тизимининг барча функцияларини фойдалирок ташкил этиш ва хисоблаш тизими ва фойдаланувчи уртасидаги интерфейсни ташкил этиш учун мулжалланган.

Изох: Факатгина тизимли бошкарувчи дастурларгина аппаратурага бевосита мурожат кила оладилар.

Изох: Тизимли бошкарувчи дастурларни операцион тизимлар (ОТ) деб аталади.

ОТ интерфейснинг куйидаги вариантларини таъминлаши мумкин:

- команда интерфейси;
- дастур интерфейси (чакириклар тизими ёки баъзи бир тизимли функцияларни бажариш учун кисм дастурлар куринишида);
- фойдаланувчи интерфейси (дарча, меню, клавишалар ва х.к.)

3. Кушимча тизимли дастурлар кайта ишловчи ва бошкарувчи дастурларни имкониятларини кенгайтириш учун мулжалланган.

Улар таркибига :

- сервис дастурлари;
- инструменталь дастурлар киради.

Сервис дастурларга:

- дастур кобиклари (надстройки);
- утилиталар киради.

Дастур кобигининг яхши томони – бу хисоблаш тизимининг захираларига мурожатни яхшилашдан иборат (Windows проводниги в ах.к.).

Утилиталар (ёрдамчи хизмат курсатувчи дастурлар) фойдаланувчини кушимча имкониятлар билан таъминлаш (архивлаш, маълумотларни тиклаш, дискларга хизмат курсатиш, вирусга карши дастурлар).

Инструменталь дастурий курилмаларга куйидагилар киради:

- СУБД;
- Машина графикаси тизимлари

ва х.к.

4. Текширув-диагностик дастурлар ЭХМ нинг ишлатиш жараёнидаги носозликларни текшириш, аниклаш ва бартараф этиш профилактикаси учун мулжалланган.

5. Амалий дастурлар пакети – бу амалий масалаларни ечиш учун мулжалланган дастурлар тупламидир. Уларга – илмий хисоблашлар, моделлаштириш ва х.к. мисол булади.

Изох: Хар бир амалий дастурлар пакетининг уз тили булиб, бу пакетга тегишли ишларнинг бажарилиш тартиби ушбу тилда ифодаланади.

7. Хужжатлар мажмуаси – матнли хужжатларнинг ГОСТ (ЕСКД) га мувофик тайёрланган туплами булиб, уларда тизимли дастурий таъминотнинг мос булакларини эксплуатация килиш ва урнатиш хакидаги маълумотлар берилади.

2.Дастурлаш тизимлари

Дастурлаш тизимлари тил муаммоларини хал килувчи дастурларни бирлаштирадилар ва дастурий таъминотни ишлаб чикаришга мулжалланган.

Дастурлаш тизимларига 1) трансляторлар; 2) кутубхона дастурлари; 3)редакторлар;4)компановщиклар; 5)загрузчиклар;6)отладчиклар кирадилар.

Дастурларга хизмат курсатувчи тизимлар – бу махсус сервис дастурлар булиб, улар ёрдамида операцион тизимни узига хизмат курсатиш мумкин.

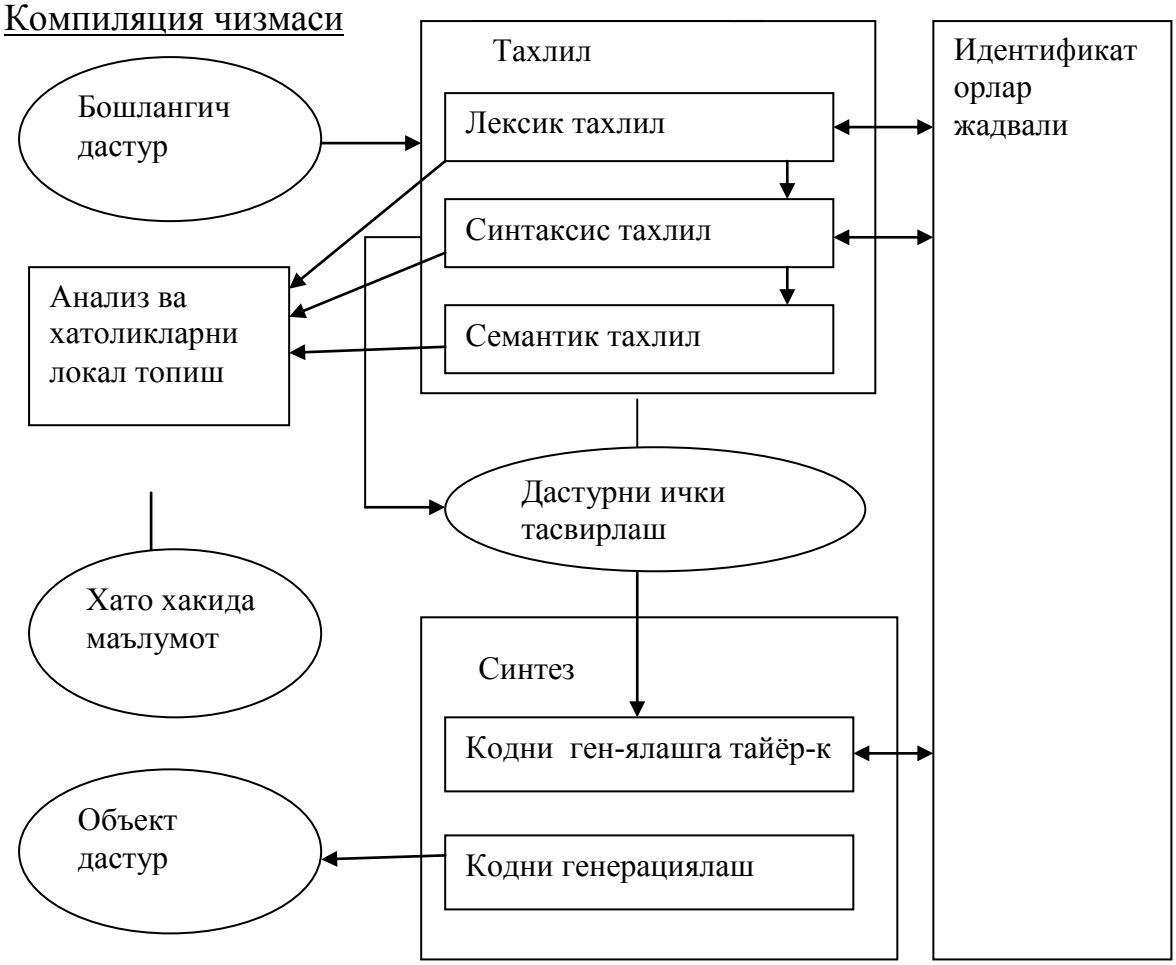
Транслятор – бу дастур берилган дастурлаш тилидаги кирувчи дастур матнини унга эквивалент булган чикишдаги натижавий тилга угиради.

Компилятор – бу транслятор булиб, у берилган дастур мантнини унга эквивалент булган машина командаларидаги объект дастурга угиради.

Интерпретатор – бу дастур булиб, у берилган дастур матнини бирданига кабул килади ва бажаради (натижавий коди булмайди)

Компилятор формал тиллар нуктаи назаридан куйидаги 2 асосий функцияларни бажаради:1) у кирувчи дастур матни тили учун англовчи хисобланади (кирувчи дастур занжирлар генератори булиб хисобланади);2)натижавий дастур тили учун генератор хисобланади (англовчи булиб хисоблаш тизими хисобланади)

Компиляция чизмаси



Лексик тахлил – бу компилятор булаги булиб, дастур литераларини укийди ва улар оркали кирувчи тил лексемаларини куради.
Синтаксис тахлил – Тахлил боскичидаги компиляторнинг асосий булагидир. Тилнинг систаксис конструкцияларини ажратади.
Семантик тахлил – бу компилятор булаги булиб, кирувчи тил семантикаси нуктаи назаридан дастур матнини текширади.
Кодни генерациялашга тайёргарлик – натижавий дастурнинг синтези билан боглик булган харакатларга тайёргарлик бажарилади.
Кодни генарациялаш – натижавий кодни бевосита хосил этиш – кодни оптимизациялашни уз ичига олган асосий фаза.
Идентификаторлар жадвали – кирувчи дастур элементлари хакидаги маълумотларни сакловчи берилганлар туплами. Бир неча хил идентификаторлар жадвали мавжуд.
Утиш – бу ташки хотирадан берилганларни охирги укиш жараёни, уларни кайта ишлаш ва ташки хотирага жойлаштириш. Компиляциянинг бир фазаси - бир утишдир.

3. Асосий функциялари ва таркиби

Тизимли кайта ишловчи дастурларнинг асосий функциялари ва таркиби;

1. Ассемблер;
2. Алока редакторлар ва юкловчилар;
3. Макропроцессорлар;
4. Трансляторлар (таржимонлар);
5. Тил конверторлари;
6. Редакторлар ва матн процессорлари;
7. Отладчиклар;
8. Дизассемблер;
9. Кросс-системлар;
10. Кутубхоначилар.

1. Ассемблер – бу шундай тизимли кайта ишловчи дастур булиб, у бирон бир машинага мулжалланган дастурлаш тилида ёзилган дастур матнини объект кодига айлантириш учун мулжалланган. (Ассемблер тилидаги матн директивалар ва исмлардан ташкил топади, машина коди эса факат байтлардан ташкил топади.).

Изох: Объект коди ёки алока редакторининг киришига, ёки юкловчининг киришига келиб тушади.

2. Алока редакторлари – бу шундай тизимли кайта ишловчи дастур булиб, улар Ассемблер ёрдамида алохида олинган объект модулларини ягона модулга бирлаштиришга мулжалланган. Алока редактори чегарасида барча адрес йуналишлари ягона адреслар фазосига урнатилади.

Алохида объект модулларида хар бир объект модули алока редакторининг чикишини юкловчининг кириши деб хисоблайди.

Юкловчилар дастурни кайта ишловчи дастурга юклайлилар ва унга бошкарувни узатадилар. Яна шу билан бирга улар юкловчиларни богловчи алохида модулларни бирлаштирадилар.

Изох: Юкловчилар жой узгартирувчи ёки абсолют булишлари мумкин.

- Абсолют юкловчилар хар бир дастурни биттадан фиксирланган адрес буйича юклайдилар.
- Жой узгартирувчи юкловчилар дастурни хотирадаги ихтиёрий буш жойга жойлаштиришлари мумкин.

3. Макропроцессорлар– бу шундай дастурларки, улар белгили кайта ишлашга мулжалланган булиб, бу жараёнда кандайдир киска фразаларга (макрочакирикларга) узун фразалар (макрокенгайтмалар) мос куйилади. Макропроцессорнинг киришида кандайдир макрочакириклардан олинган матн булиб , чикишида эса – макрокенгайтмалар булади.

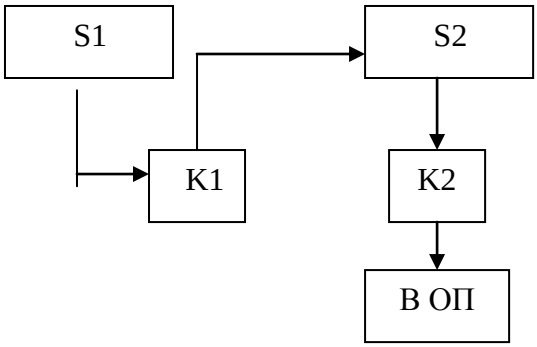
4. Трансляторлар (таржимонлар) бир тилда ёзилган матнни бошка тилга угирадилар. Трансляторларнинг куйидаги куринишларини ажратиб курсатиш мумкин:

- компиляторлар: киришида юкори даража тилида ёзилган дастур матни, чикишида машина кодларидаги алока редакторига ёки юкловчига узатиладиган дастур.

Хусусиятлари: таржима функцияларини аник булиниши ва таржима килинган функцияларни бажарилиши..

- интерпретаторлар – функциялар булинмайдилар, балки мослаштириладилар. Интерпретатор таржимани ва бажарилишни каторлаб ва кооператив бажаради. Улардан ёзилган дастурни диалог асосида кайта ишлашда фойдаланиш кулай.

5. Тил конверторлари бир юкори даража дастурлаш тилида ёзилган дастур матнини бошка юкори даража дастурлаш тилига айлантириш учун мулжалланган. Улар S1 дастурлаш тилида ёзилган дастурни S2 дастурлаш тилига айлантириш учун керак..



6. Матн редакторлари мантни кайта ишлаш учун кенг имкониятларга эгаликлари билан фаркланадилар.

7. Отладчиклар дастурларни бажарилиш жараёнида учраши мумкин булган хатоларни кидириш ва бартараф этиш учун мулжалланган.

8. Дизассемблер бу машина кодлари кетма-кетлигини ассемблер куринишига узгартирадиган дастур.

Изох: Улар хам баъзи бир харакатлар тизимини бажарилишини ассемблер куринишида куриш имконини яратадилар.

9. Кросс-тизим – бу дастур бир хисоблаш машинасида машина кодларида ифодаланган бошка бир хисоблаш машинасининг дастурларини олиш учун кулланилади. Лойихалаштирилаётган хисоблаш тизимлари архитектурасини отладка килиш учун фойдаланилади.

10. Кутубхоначилар – киритилаётган матн , объект модули раками булиши мумкин булган кутубхона файлларини ташкил этиш ва уларга хизмат курсатиш учун дастурлардир.

Операцион тизим ривожланиш имкониятлари:

Операцион тизим узлуксиз ривожланишининг сабаблари:

1. Аппарат таъминотининг янги куринишларининг юзага келиши ва янгиланиши.

2. Янги сервислар.

3. Узгартиришлар киритиш.

Операцион тизимнинг ривожланиши зарурият, акс холда у янги дастурлар ва янги курулмалар билан ишлай олмайди.

Мисол учун, Intel процессорлар Hyper Thread ни кулланишидан бошлаб Windows NT операцион тизимини бозордан сикиб чикара бошлади, улар иккинчи процессорни эмуляция килдилар. Как работать с эмуляцией

ядра Windows NT ни ядросини эмуляцияси билан кандай ишлашни «билмай колиб» тизим йикилди.

Операцион тизимлар ривожланиши мумкин.

1.сакраб (Windows) ва

2.аста секин (UNIX).

Биринчи холда аввалги мавжуд дастурлардан тулик воз кечишга тугри келади, чунки бу тизимда хеч кандай кучириб утишлар назарда тутилмаган. Аммо, факат унинг узигагина ёзилган янги дастурлардан бутун махсулот юзага келади. Бу эса дастурий таъминотни ишлаб чикарувчилар (яратувчилар) учун жуда фойдалидир.

Иккинчи холда янги курилма имкониятларини эски дастурлар билан боглашга тугри келади. Операцион тизим ва дастурлар имкониятларини кенгайтирувчи дастур заплаткаларини (патчлар) ишлаб чикаришга тугри келади.

Операцион тизимларни ишлаб чикиш.

Операцион тизимларни ишлаб чикиш буйича уч ечиш йули мавжуд.

Ёпик ечим. Барча ишлаб чикарилаётган махсулотлар лицензиялар билан химояланган ва ишлаб чикарилган фирмадан ташкарида узгартирила олмайди. Бундай ечим фирмадан курилмаларнинг узгаришига жуда тез эътибор беришни, маркетинг ва ишлаб чикаришга катта маблагларни сарфлашни талаб килади.

Бундай йуналишнинг ютуги булиб, узи ишлаб чикараётган дастурий махсулоти учун фирманинг жавобгарлиги хисобланади. Яхши хужжатлаштириш. Универсальлик. Стандартлаштириш. Бундай йуналишдан фойдаланаётган таникли фирмалардан : Microsoft, SUN, SCO.

Бундай йуналишнинг камчилиги фирмаларнинг инертлиги хисобланади. Узгараётган шароитларга тезда ахамият бера олмаслик кобилияти. Операцион тизимнинг бошка фирма ишлаб чикараётган операцион тизим билан узаро алока кила олмаслиги, яъни мослаша олмаслиги имконияти.

Очик ечим. Ишлаб чикарилаётган ечимлар умумий лицензияга эга очик кодли ечимга буйсунадилар. Ихтиёрий хохловчи дастур махсулотининг берилиш кодини олиши мумкин, агар унинг автор томонидан ишчи варианти куйилган булса. Бундай ечим дастурий махсулот хатосиз ишлашига кафолат бера олмайди. Бу ечимлар куп холларда яхши хужжатлаштирилмаган. Бундай йуналишнинг ютуги булиб, турли давлат ва турли соха мутахассисларининг бир командада ишлай олиши имконияти хисобланади. Ишлаб чикарувчилар командасининг узгариш шартларига тезликдаги эътибори. Барча мумкин булган платформаларда ва ихтиёрий бошка тизимларда ишлай олиш имконияти. Бундай йуналишдан фойдаланаётган таникли фирмалардан: RedHat, SuSe, FreeBSD; Novell.

Аралаш ечим. Ишлаб чикарилаётган ечимлар умумий коддан ташкари яна шахсий ишлаб чикарилаётган, лицензия билан химояланган ечимдан хам фойдаланадилар. Бундай йуналиш очик ечимлардан энг яхши ечимларни танлаб олиш ва улар асосида узларининг ечимларини ташкил этиш имконини

беради. Бундай йуналиш хар иккала йуналишнинг энг яхши хусусиятларини хисобга олади, чунки фирма факатгина узининг ечимларинигина хужжатлаштириш ва унга жавобгарликни буйнига олибгина колмай, балки танлаб олинган очик ечимлар учун хам жавобгарликни хис этади. Бундай йулни танлаган фирмалар MacOS, BeOs, QNX, Netrino.

Назорат саволлари

1. Хисоблаш тизимини кандай булаклар ташкил этади?
2. Амалий дастурий таъминотнинг вазифаси нималардан иборат?
3. Тизимли дастурий таъминотнинг вазифаси хакида тушунча беринг.
4. Дастур интерфейси нима учун керак ?
5. Фойдаланувчи интерфейси нима учун керак?
6. Дастур кобиклари нима?
7. Утилиталар нима?
8. Тизимли кайта ишловчи дастурларнинг асосий функциялари ва таркиби хакида маълумот беринг.
9. Операцион тизим нима?
10. Операцион тизимларни ишлаб чикишдаги уч ечиш йуналишлари хакида маълумот беринг.

Фойдаланилган адабиётлар

1.Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
2.Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
3.Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
4.Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
5.WWW.codecrojekt.ru
6.WWW. master.ru
7.WWW.bdn_borland.com
8.http://microsofft.com

Маъруза № 2.
Мавзу. Файллар тизими.
Режа:
1.Файлларнинг асосий хусусиятлари.
2.Берилганларни химоялаш.
3.Файллар тизимининг асосий хусусиятлари.

Калит сузлар.
- **Файл**
- **Файллар тизими**
- **Файл кўрсаткичи**
- **Шахсийлаштириш**
- **Операцион тизим**
- **Узлуксиз сегментли файллар**
- **Блокли ташкил этилган файллар тизими**
- **Иерархик файллар тизими**

Биз аввал айтиб утганимиздек, хар бир операцион тизим жараён деб аталувчи тушунчага эга. Яна иккинчи бир тушунча хам борки, у хам жуда мухим булиб –бу файлдир. *Файл тизими - бу операцион тизим компонентаси булиб, номланган берилганлар тупламига мурожаатни, ташкил этиш ва саклашни таъминловчидир.Берилганларнинг номланган туплами файллар деб аталади.*

1.Файлларнинг асосий хусусиятлари.

1. Файл -бу исмга эга объект булиб, шу исм оркали файлни ичидаги маълумотлар билан ишловчи объектдир. Исм бу белгилар кетма-кетлиги булиб, унинг узунлиги аник операцион тизим турига богликдир.

2. Файлни жойлашишига боглик эмаслиги. Аник бир файл билан ишлаш учун у файлнинг ташки курилмадаги жойлашишини билиш талаб килинмайди.

3. Кириш/чикиш функциялари туплами. Хар бир операцион тизим файллар билан маълумот алмашинувни таъминловчи функциялар тупламига эга. Бу функциялар туплами куйидагилардан ташкил топади:

1. Файл иш учун очилган. Ёки мавжуд ёки янги файлни очиш мумкин. Шундай савол тугилиши мумкин. - нима учун файлни очиш керак? Нима учун бирданига файлдан укиш ва файлга ёзиш мумкин эмас? Хакикатда, бу операцион тизимга файл аник жараён билан ишлашини марказий равишда эълон килиш воситасидир. У эса ушбу маълумотларга асосан кандайдир ечим кабул килиши мумкин. (масалан, бошка жараёнлар учун ушбу файлга мурожаатни чеклаб куйиши мумкин.).
2. Укиш/ёзиш. Купинча файл билан маълумот алмашинув берилганлар блоки куринишида ташкил этилиши мумкин. Ушбу берилганлар блоки икки хил хусусиятга эга. Бир томондан ихтиёрий хисоблаш тизими учун берилганлар блокининг улчами

аник берилган, яъни булар аппарат -дастур улчамларидир. Иккинчи томондан бу берилганлар блоки реал алмашинувда дастурчи томонидан ихтиёрий равишда бошкарилиши мумкин. Укиш/ёзиш функцияларида купинча алмашинув учун берилганлар блоки улчами ва укилиши ёки ёзилиши керак булган берилганлар блоки сони берилади. Танланган берилганлар блокининг улчамидан алмашинувларнинг унумдорлиги боглик, фараз килайлик бир машина учун берилганлар блокининг унумдорлик улчами 256 Кб булса, сиз 128 Кб лик алмашинувни амалга оширмокчи булсангиз, у холда сиз мантикий блокларни укиш учун икки маротаба 128 Кб дан мурожаат киласиз. Бу холда сиз 256 Кб ни бир мартада укиш урнига, бир блокга икки маротаба мурожаат киласиз ва бир сафар ярмини, кейинги сафар кейинги ярмини укийсиз. Бу ерда яна баъзи бир «унимсизлик» элементлари учраши хам мумкин, лекин уларни «аклли» операцион тизим текислаб юборади, агар текислай олмаса, демак бу сизнинг хатойингиз булади.

3. Файл курсаткичини бошкариш. Хар бир очилган файл билан файл курсаткичи тушунчаси боглик. Бу курсаткич командаларнинг хисоблагич регистри булиб, хар бир вактда кейинги файл буйича алмашинувни амалга ошириш мумкин булган нисбий адресни курсатади. Ушбу блок билан алмашинув тугагандан сунг курсаткич блокдан ташкарига кучирилади. Файл билан ишни ташкил этиш учун ушбу файл курсаткичини бошкариш талаб этилади. Файл курсаткичини бошкариш функцияси мавжуд булиб, курсаткични файл буйича ихтиёрий (мумкин булган чегараларда) кучириш имконини беради. Курсаткич бу кандайдир узгарувчи булиб, дастурдан мурожат килиш мумкин, ва у файлни очиш функцияси (ушбу узгарувчини ташкил этувчи) билан боглик.
4. Файлни ёпиш. Бу амал иккита функция оркали амалга оширилиши мумкин:
   1) Файлни ёпиш ва охирги кийматини саклаб колиш.
   2) Файлни йукотиб (учириб) ташлаш.

   Файл ёпилгандан сунг у билан барча алокалар тугатилади ва у каноник холатга утади.

2. Берилганларни химоялаш. Купгина стратегик ечимлар аппарат даражасидаги каби, операцион тизим даражасида хам такрорланади. Агар биз мультипрограмма режимини эсласак, унинг мавжуд булишлигининг зарурий шартларидан бири бу хотира ва берилганларни химоялашни таъминлашдан иборат эди. Агар биз файллар тизимини карасак, у хам худи операцион тизим каби бир фойдаланувчилик булиши мумкин. Бу холатда берилганларни химоялаш муаммоси булмайди, чунки операцион тизимда ишлаётган киши барча файлларни эгаси хисобланади. MS-DOS ёки Windows 95 бир

фойдаланувчилик тизимларга мисол булади. Машинани юклаб бошка фойдаланувчиларнинг барча дисклардаги файлларини учириб ташлаш мумкин, чунки бу тизимларда хеч кандай химоялаш йук. Куп фойдаланувчилик тизим купгина фойдаланувчиларнинг тиник ишлашини таъминлайди. MS-DOS операцион тизими хам мультипрограмма режимида ишлаши мумкин, лекин жуда хам тиник эмас, бир жараёндаги хатолик операцион тизимнинг ва кушни жараённинг учирилишига олиб келиши мумкин. Худди шундай Windows 95 операцион тизимида хам купфойдаланувчи ишлаши мумкин, лекин унинг иши хам тиник эмас, чунки операцион тизим уланрнинг барча хукукларини химоясини таъминламайди. Шундай килиб, купфойдаланувчилик тизим ташки таъсирлардан химоя килиши керак. Аслида химоялаш муаммоси факатгина файл тизими билан боглик эмас. ОТ барча сохаларда берилганларни химоялашни таъминлайди: бу файллар, жараёнлар, жараёнларга тегишли бир фойдаланувчи томонидан куйилган ресурслар. Бу ерда мен сизнинг эътиборингизни шу фактга каратаман, чунки файллар учун бу жуда мухим нукта.

3.Файллар тизимининг асосий хусусиятлари.

Файллар тизими файллар учун санаб утилган барча хусусиятларни уз ичига олади, ва яна баъзи бир кушимча хусусиятларга хам эга. Бу хусусиятлар файллар тизимининг структуралик ташкил этилиши билан боглик.

Келинг, кандайдир ташки саклаш курилмаси (ВЗУ) фазосини караб чикайлик ва бу фазода файлларни жойлаштиришни ташкил этишни куриб чикамиз.

1. Узлуксиз сегментли файлларни бирдаражали ташкил этиш. «Бирдаражали» **т**ермини тизим уникал номланган файллар билан ишлашни таъминлашини англатади. Ташки саклаш курилмаси чегарасида берилганларни саклаш учун каталог деб аталувчи соха ажратилади. Каталог куйидаги структурага эга:

исм	Бошлангич блок	Охирги блок

«Бошлангич блок» берилган исм билан бошланувчи ташки саклаш курилмасидаги нисбий адресни курсатади. «Охирги блок» берилган файлнинг охирги блокини аниклайди. Файлни очиш функцияси каталогда файл исмини топиш, унинг бошланиши ва охирини аниклашни амалга оширади. (амалда берилганлар курсатилганидан кам жой эгаллаши мумкин, лекин бу хакда кейинрок тухталамиз). Бу харакат жуда оддий, шу билан каталогни ОТ хотирасида саклаш мумкин, бу эса алмашинувларни камайишига олиб келади. Агар янги файл ташкил этилаётган булса, у буш

жойга ёзилади. Исмлар каталогига ухшаш буш фазолар (фрагментлар) жадвали булиши мумкин.

Укиш/ёзиш кушимча алмашинувларсиз амалга оширилади, чунки файлни очишда биз берилганларни жойлаштириш диапазонига эга буламиз. Укиш ушбу структура блокига мос равишда амалга оширилади ва хеч кандай кушимча маълумот талаб этилмайли, алмашинув хам мос равишда тезда амалга оширилади.

Энди караб чикайлик, бундай файлга кушимча маълумот ёзмокчимиз, лекин буш фазо жой йук? Бу холда тизим икки хил йул тутиши мумкин. Биринчидан, у сизга жой йуклигини айтади ва сиз узингиз нимадир килишингиз керак булади, яъни кандайдир Ушбу файлни бирор жойга кучириб турадиган ва керакли маълумотни кушадиган жараённи куясиз. Бундай кучириш етарли даражада кимматга тушадиган функция. Иккинчи имконият - алмашинувни рад этилади. Бу эса файлни очиш жараёнида аввалдан кушимча жой олиб куйиш кераклигини англатади; бу холда файл тизими буфернинг буш улчамини текширади ва у кам булса, Ушбу файлни жойлаштириш учун буш жой кидиради.

Бундай курамизки, бундай ташкил этиш сода, алмашинувларда унумли, лекин файл учун жой етишмаган холларда унимсизлик бошланади. Бундай ташкари файл тизимининг узок ишлаши давомида дискда худди оператив хотирадаги каби фрагментация холати юз беради. Яъни буш жойлар мавжуд ,лекин файлимизни жойлаштириш учун етарли жой йук булган холат юзага келади. Файл тизимини бундай ташкил этилишининг фрагментацияси билан курашишда узок, огир ва файл тизими учун хавфли булган жараён, яъни файлларни бир-бирига зичлаштириш жараёни амалга оширилади.

Бундай ташкил этиш бир фойдаланувчилик файл тизими учун кулай ва фойдалидир, чунки фойдаланувчиларнинг куплиги холатида фрагментация юз беради. Зичлаштириш жараёнини хар доим куйиш максадга мувофик эмас. Бошка томондан тизим оддий ва хеч кандай кушимча харажатлар талаб килмайди.

2. Файллар блокли ташкил этилган файллар тизими. Ташки саклаш курилмалари фазоси блокларга булинган. Файллар тизимида бундай маълумотларни булаклашда оператив хотирани варакли ташкил этишдаги жараёнлар маълумотларини булаклаш каби амалга оширилади. Умумий холда, хар бир файл исми билан шу файл берилганлари жойлашган курилма блоклари ракамларини туплами боглик. Ушбу блокларни ракамлари ихтиёрий тартибга эга, яъни блоклар курилма буйича ихтиёрий таркалган. Бундай ташкил этишда фрагментация муаммоси йук, аммо блокни яхлитлаш йукотишлари мавжуд (агар файл блокни битта байтини банд килган булса, у холда бутун блок банд хисобланади). Шундай килиб, зичлаштириш муаммоси йук, ва бу тизим купфойдаланувлилик ташкил этишда фойдаланиш мумкин.

Бу холда хар бир файл бир канча атрибутлар туплами билан боглик: файл исми ва файлга мурожаат этиладиган фойдаланувчи исми; Бундай

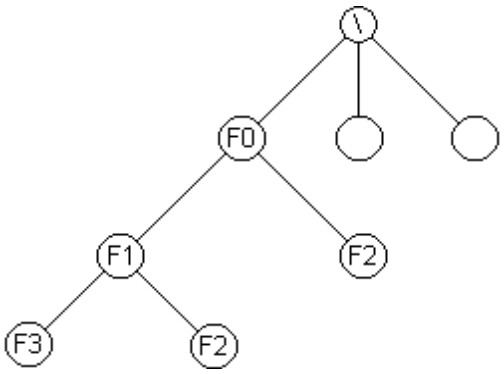
ташкил этилиши исмларни уникаллиги муаммосидан кутилишга имкон беради. Бундай тизимда исм уникаллиги бир фойдаланувчилик файллар уртасида талаб этилади.

Бундай файлларни ташкил этиш каталог оркали амалга оширилади. Каталогни структураси куйидагича булади. Каталог каторлардан ташкил топади; хар бир i-чи катор файл тизимини i-чи блокга мос келади. Бу каторда блок банд ёки бушлиги хакидаги маълумот сакланади. Агар у банд булса, у холда бу каторда файл исми (ёки унга мурожаат), фойдаланувчи исми ва бошка кушимча маълумотлар жойлашиши мумкин.

Маълумот алмашинув даврида тизим турлича харакатланиши мумкин. Ёки файлни очишда тизим каталог буйича айланиб файлни мантикий блокларини дискда жойлашиш жадвалини куради. Ёки хар бир алмашинувда бу мослик амалга оширилади.

Файллар тизимини бундай ташкил этилиши бир фойдаланувчи рамкасида бирдаражали хисобланади, яъни барча файллар кандайдир фойдаланувчига тегишли гурухга богланган.

3. Иерархик файллар тизими. Файл тизимининг барча файллари дарахт деб аталган бир структурага курилган. Дарахтнинг илдизида файл тизимининг илдизи жойлашган. Дарахтни богланган жойи варак хисобланса, бу файл фойдаланувчининг берилганларидан ташкил топиб файл-каталог хисобланади. Дарахтнинг варакдан фаркли богланишлари файл-каталоглар хисобланади. Бундай иерархик файл тизимида номланиш турли усуллар билан амалга ошади. Биринчи тур – файлни энг якин каталога нисбатан номлаш, яъни биз F0 каталог учун якин булган файлларни карасак, бу F1 файл булиб, у хам каталогдир ва F2 файл. Бундай тизимда бир даражада исмлар такрорланмаслиги максадга мувофик. Бошка томондан, барча файллар дарахт билан богланганликлари учун биз файлни тулик номи, яъни файл тизими илдизидан аник бир файлгача булган йул, хакида гапира оламиз. F3 файлнинг тулик исми куйидагича белгиланади: /F0/F1/F3. Бундай ташкил этиш файлнинг киска исми билан хам, тулик исми билан хам ишлаш имконини беради. Файлларнинг тулик исми бу йулдир. Ихтиёрий дарахтда унинг илдизидан ихтиёрий богламигача бита йул мавжуд, шундай килиб исмларни унификация килиш муаммоси хал этилади. Биринчи марта бундай усул Беркли университетида 60-йилларнинг охирида ишлаб чикилган Multix операцион тизимида фойдаланилган. Кейинчалик бу чиройли ечим купгина операцион тизимларда кулланила бошлади. Бу иерархияга мос равишда хар бир файлга кандайдир мурожат хукукига эга атрибутларни боглаб куйиш мумкин. Мурожаат хукукларига фойдаланувчилар файллари билан биргаликда каталоглар хам эгадирлар.Бу тизимнинг структураси купфойдаланувчилик ишни ташкил этишда, исмлар муаммосини йуклиги хисобига унумдордир.



4. Операцион тизимда шахсийлаштириш ва берилганларни химоялаш. Бу нюанс, хам сода, хам мураккаб. Соддалиги шундаки биз у хакида бир неча огиз гапирамиз холос, мураккаблиги шундаки, шундай муаммолар мавжудки улар хакида узок гапириш мумкин.

Шахсийлаштириш – бу аник фойдаланувчини идентификация кили шва шу билан мос равишда берилганларни химоялаш буйича у ёки бу харакатларни кабул килиш имкониятидир.

Агар биз ихтиёрий MS-DOS операцион тизимини карасак, у бир фойдаланувчилик.

Операцион тизимларнинг иккинчи даражаси – фойдаланувчиларни руйхатдан утказадиган, лекин барча фойдаланувчилар ягона субъект туплами куринишида ва бир-бири билан боглик эмаслар. Бундай операцион тизимларга мисол сифатида IBM фирмасининг mainframe-компьютерлари учун операцион тизимларини курсатиш мумкин. Мисол учун маърузачи узининг эшитувчи талабаларнинг кайси бири кандай гурухга тегишли эканлигини билмайди, лекин уларнинг шу крс талабалари эканлигини билади. Бу хам яхши, хам ёмон. Бу курсни эшитиш учун яхши, лекин маърузачи томонидан савол жавоб килиш масаласида ёмон, чунки бир кун ичида у хамма талабалар билна савол-жавоб килишга улгура олмайди. Шунинг учун у барча эшитувчиларни кандайдир булаклаши керак, лекин кандай бу ноаник.

Мос равишда бундай бир улчамли шахсийлаштиришда барча биз айтиб утган функциялар (хусусан, химоялаш) таъминланади, лекин фойдаланувчиларни бундай ташкил этиш фойдаланувчилар гурухини англатмайди. Менга эса бизнинг факультет серверида мениг лабораториям ажратилса ва бу лаборатория рамкасида файлларга мурожаат хукукини бир бирига бериш имкони берилса максадга мувофик булар эди.

Мос равишда файллар тизимидаги каби фойдаланувчиларнинг иерархик ташкил этиш пайди булади. Яъни бизда «барча фойдаланувчилар» ва «фойдаланувчилар гурухи» деган тушунча мавжуд. Гурухда реал фойдаланувчилар мавжуд. Бундай шахсийлаштиришни иерархик ташкил этиш куйидаги ларни келтириб чикаради. Кандайдир фойдаланувчини руйхатдан утказиш учун уни аввал кандайдир гурухга киритиш керак, - бу лаборатория, кафедра ёки укув булимии булиши мумкин. Фойдаланувчилар гурухларга бирлашганликлари учун, фойдаланувчиларнинг ресурсларига мурожаат хукукини булиниш имконияти юзага келади. Яъни, масалан фойдаланувчи унинг ресурсларидан барча гурухдаги фойдаланувчилар фойдаланишлари мумкин эканлигини эълон килиши мумкин. Бундай чизма купдаражали (гурухлар гурухчаларга булинадилар ва х.к.) мос хукук ва имкониятлардан келиб чиккан холда булиши мумкин. Хозирги кунда шундай операцион тизимлар яратилмокдаки, уларда мурожаат хукуки факатгина иерархик структурага боглик булиб колмай, балки мураккаброкдир, яъни мурожаат хукукини иерархияни бузган холда кушиш мумкин.

Назорат саволлари.

1. Файл нима ?
2. Файлларнинг асосий хусусиятлари хакида маълумот беринг.
3. Файл курсаткичи нима?
4. Файллар тизими нима ?
5. Файллар тизимининг асосий хусусиятлари хакида маълумот беринг.
6. Узлуксиз сегментли файлларни бирдаражали ташкил этишнинг ютук ва камчиликлари хакида маълумот беринг.
7. Файллар блокли ташкил этилган файллар тизимининг ютук ва камчиликлари хакида маълумот беринг.
8. Иерархик файллар тизими тизимининг ютук ва камчиликлари хакида маълумот беринг.

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза №3.

Мавзу: Берилганлар структураси ва уларни тасвирлаш ва улардан фойдаланиш усуллари.

Режа:

1. **Берилганлар структураси тушунчаси ва уларни умумий ифодалаш;**
2. **Берилганлар структурасини классификацияси;**
3. **Оддий статик берилганлар структураси.**

Калит сузлар.

- **мантиқий структура**
- **физик структура**
- **Статик**
- **Яримстатик**
- **Динамик**
- **чизикли-тартибланган**
- **чизикли-тартибланмаган**

1.Берилганлар структураси тушунчаси ва уларни умумий ифодалаш Хисоблаш жараёни берилганлар дастури ёрдамида амалга оширилади, дастурнинг узи эса специфик табиатга эга берилганларни ифодалайди. Берилганлар:

1. мантикий даражада;
2. физик даражада (машина даражасида), берилганлар физик хотирада ифодаланади.

Иккала даражада хам берилганларга аник таъриф бериш мумкин.

Мантикий даражада берилганлар кандайдир информацион объектлардан ташкил топган булиб, улар кандайдир фактларни канайдир алгоритм ёрдамида кайта ишлаш максадида формал куринишда ифодалашга хизмат киладилар.

Физик даражада мураккаблиги ва мазмунидан катьи назар берилганлар иккилик разрядлари кетма-кетлиги куринишида ифодаланади. Бу даражада берилганлар узининг ички табиатига эга эмас, яъни улар структураланмаган.

Биргаликда караладиган ва кайта ишланадиган элементар берилганларнинг туплами **берилганлар структураси** деб аталади.

Умумий холда берилганлар структураси – **берилганлар элементлари** уртасида мавжуд муносабатларни ифодаловчи коида ва чегаралар тупламидир.

Бир берилганлар элементининг бошка берилганлар элементи барча алокалари кандайдир берилганлар структурасида **«муносабат» элементи**ни ташкил этади.

Берилганлар структураси бир-бири билан узаробогланган берилганлар элементларидан ташкил топади. Структуранинг хар бир элементи умумий холда 2- булакдан ташкил топади:

1. Берилганлар элементи;

2. Муносабат элементи.

Уз навбатида берилганлар элементи кандайдир турдаги **бир кийматли берилганлар** булиши мумкин, ёки бир турдаги кандайдир **берилганлар туплами** булиши мумкин.

Берилганлар элементи турлари тушунчасига куйидагилар киради:

1. мумкин булган кийматлар диапазони (сохаси);
2. мумкин булган амаллар туплами;
3. машинада ифодалаш усули.

Берилганлар структураси элементлари орасидаги узароалока граф куринишида ифодаланиши мумкин, бу ерда чуккилар берилганларнинг алохида элементлари, кобиклар эса улар орасидаги алокалар.

Мантикий даражада берилганлар структураси **мантикий (абстракт) структура** деб аталади, чунки берилганлар хотирада кандай жойлашганлигига боглик эмас.

Физик даражада берилганлар структураси **физик (саклаш структураси) деб аталади.**

Мантикий ва физик берилганлар структураси уртасида жуда катта фарк мавжуд, масалан, дастурчи нуктаи назаридан, туплам (икки улчамли) жадвал куринишига эга булиб, ЭХМ хотирасида берилганлар элементларининг чизикли кетма-кетлигини ифодалайди.

Берилганларни кайта ишлаш тизимида хар доим махсус процедуралар ишлаб чикилади ва улар мантикий даража берилганларини физик даража берилганларига айлантирадилар. Бу процедураларнинг бош максади – мантикий даражадан келиб чиккан холда физик даражадаги берилганларга мурожаатни таъминлаш. Масалан, икки улчамли туплам учун шундай акс эттириш функцияси мавжуд булиши керакки, у туплам элементининг жадвалдаги координатасидан келиб чиккан холда аник манзилини олиш имконини берсин.

$A_{ij} = A_0 + a(i, j); \quad a(i, j) - \text{функция}$

Мантикий ва физик куринишдан ташкари ихтиёрий структура мантикий ва физик структуралар амаллари туплами билан характерланади. Физик структуралар устидаги амаллар туплами кандай хотирада амалга оширилаётганига кура аникланади: оператив хотирадами ёки ташки хотирадами?

Оператив хотирадаги холатнинг хусусияти шундаки у хотира ячейкаларининг тартибланган кетма-кетлигидан ташкил топган булиб, хотиранинг ихтиёрий ячейкасига тугридан тугри мурожаат этиш имкониятига эга, кетма-кет ракамлангандир.

Ташки хотирада эса берилганлар структураси турли турда ташкил этилган хисобланади (кетма-кет, тугридан -тугри, индексли-кетма-кет).

2.Берилганлар структурасининг классификацияси

4 та белги буйича :

1. структура элементлари орасидаги алокаларнинг борлиги ёки йуклиги. Бу белги буйича ажратилади:

- богланган структуралар;

- богланмаган структуралар.

Богланмаган структураларга векторлар, тупламлар, каторлар киради. Богланмаганларига барча мумкин булган руйхатлар киради.

2. структуранинг узгарувчанлиги нуктаи –назаридан ажратилади:

- статик. Статик – структураларда элементлар сони ва элементлар уртасидаги алокалар сони узгармайди (каторлар, векторлар, тупламлар);
- яримстатик. Яримстатик – структуранинг баъзи булаклари узгариши мумкин (стеклар);
- динамик. Динамик – структураларда элементлар сони ва элементлар уртасидани алокалар сони ихтиёрий вактда узгариши мумкин (руйхатлар).

3. тартибланиш даражаси буйича фаркланади:

- чизикли-тартибланган – бир элемент кейингиси кетидан каътий келади
- чизикли-тартибланмаган – чизиксиз

4. элементларнинг хотирада бир бирига нисбатан узаро жойлашиш характери. Фаркланади:

- элементлари кетма-кет жойлашган структуралар (векторлар, тупламлар, каторлар)
- элементлари ихтиёрий жойлашган структуралар. Тартиби ихтиёрий, лекин элементлари бир бири билан богланган.

3.Берилганларнинг оддий статик структуралари

Уларга

1. векторлар;
2. тупламлар;
3. ёзувлар;
4. жадваллар киради.

1. Вектор – бир турдаги берилганларнинг тартибланган туплами.

Векторга бир улчамли туплам мос келади. Унинг элементлари хотирада катъи бири-биридан кейин жойлашади. Бундай тартибланиш элементларни ракамлаш имкониятини беради (ракамлар-индекслар).

Энг куп кулланиладиган амаллардан бири – берилган индекс буйича элементга мурожаат амалидир.

Векторлар устидаги яна бошка амаллардан векторнинг пастки ва юкориги чегараларини аниклаш, вектор элементи турини аниклаш кабилардан фойдаланилади.

Векторнинг физик куриниши хотиранинг бир хил улчамдаги булаклари (майдон, слотлар) кетма-кетлиги куринишида амалга оширилади. Хар бир майдонда битта элемент сакланади. Бу слотлар хотирада элемент индекслари ошиб бориши тартибида жойлашади.

Векторнинг физик структураси хотиради жойлашган ва куйидаги табиатга эга дискрептор билан кузатилади. Дескриптор куйидаги майдонлардан ташкил топади:

- берилганлар структурасининг ички коди
- векторнинг исми
- биринчи элементнинг манзили
- индекснинг пастки киймати
- индекснинг юкори киймати
- элементнинг тури
- слот улчами

Биринчи майдон дискрептор билан богланган берилганлар структурасини аниклашга имкон беради.

Манзил функцияси (акс эттириш функцияси) вектор холатида куйидаги куринишга эга булади: $adr(i) = Adr(p) + (i - p) * I$

Бошлангич берилганлар:

- индекснинг минимал киймати: p
- i манзил кидирилаётган индекс
- максимал киймат: q
- слотнинг улчами: I

2. Туплам, умумий холда, - бух ар бир элементи вектор булиши мумкин булган берилганлар структурасидир.

Икки улчамли, учулчамли, n-улчамли тупламлар булиши мумкин. Тупламнинг хар бир элементини аниклаш учун индексдан фойдаланилади (хар бир координата буйича): $A(j_1,..., j_n)$.

Тупламнинг элементига мурожаат килиш учун n индексдан фойдаланилади

Тупламлар билан ишлаганда энг куп фойдаланиладиган амаллардан бири манзил функциясини амалга оширишни талаб этадиган мурожаат амалидир. Бу амал вектордан фаркли равишда нетривиал бажарилади.

Мисол караймиз.

Фараз килайлик n-улчамли туплам берилган булсин:

$B(i_1K_1,...,i_nK_n)$

$B(i_1,i_2...,i_n)$ – бошлангич элемент

$B(j_1,..., j_n)$ – хотирада манзилини топиш керак булган ихтиёрий элемент.

Манзил функцияси куйидагича хисобланади:

$$Adr(B(j_1,..., j_n)) = Adr(B(i_1,i_2,...,i_n)) + (\sum(j_m * D_m)) * L - (\sum(i_m * D_m)) * L,$$

бу ерда:

L – слот улчами

D – тупламни хотирада (каторлар ёки устунлар буйича) жойлашишига караб аникланади. Агар каторлар буйича жойлашган булса, у холда аввал биринчи катор жойлашади ва х.к., агар устунлар буйича жойлашса- аввал биринчи устун жойлашади ва х.к.

Элементлар катор буйича жойлашган холл учун, у холда:

$D_m = (K_{m+1} - i_{m+1} + 1) * D_{m+1}$

$m = n - 1,...,1$

$D_n = 1$

Яна шу каби учраб туради:

- симметрик квадрат матрица (элементлари бош диоганалга нисбатан симметрик ва тенг)
- кесилган матрица -разреженная матрица (купгина элементлари 0 га тенг). Бундай матрицалар руйхат куринишида сакланади.

3. Ёзув – умумий холда турли турдаги тартибланган элементларнинг тупламидир (майдонлар деб аталади).

Ёзувлар оддий ва бир-бирига киришган булиши мумкин (майдон урнига бошка ёзув жайлашиши мумкин). Бири-бирига киришиш даражаси ёзувлар дескрипторида акс этади.

4. Жадвал – бир турдаги ёзувларнинг тартибланган кетма-кетлигидир (ёзувлар туплами).

Жадваллар тизимнинг ички ифодаланишида кенг кулланилади (жадваллар хизмати, амаллар кодлари жадваллари ва х.к.).

Жадвал ёзувлардан ташкил топади, ёзув эса майдонлардан.

Ёзувларнинг бир турдалилик хусусияти барча бир исмли майдонлар бир хил турга эга эканлигини англатади.

Хар бир ёзувда жадвал чегарасида кидирувни осонлаштириш учун майдонлардан бири ключли хисобланади (барча ёзувлар ключли майдоннинг турли кийматларига эга) Ёзув хотира ячейкасининг чизикли кетма-кетлиги куринишида амалга оширилади. Хотира сохаси улчови ёзувлар сони ва хар бир ёзув регистри билан аникланади.

Жадваллар билан ишлашдаги типик амаллар:

- ёзувни кидириш (барча ёзув кидирилади);
- элементни кушиш;
- элементни учириш;
- сортировка.

Жадваллар билан ишлашда ушбу жадваллар билан амаллар бажарилиш тезлиги катта ахамиятга эга. Кичикрок жадваллар булган холида муаммолар юзага келмайди, катта жадваллар билан ишлашда эса кидирувнинг турли усулларидан фойдаланилади.

Назорат саволлари.

1.Берилганлар структураси деганда нима тушунилади?

2.Берилганларни мантикий даражада ифодалаш тушунчаси нимани англатади?

3.Берилганларни физик даражада ифодалаш нимани англатади?

4.Берилганлар элементлари нима?

5.Берилганлар структурасининг классификацияси хакида маълумот беринг.

6.Богланган структуралар нима?

7.Богланмаган структуралар нима?

8.Статик, яримстатик ва динамик структуралар хакида маълумот беринг.

9.Берилганларнинг оддий статик структуралари хакида маълумот беринг.

Фойдаланилган адабиётлар

1. Вирт Н. Алгоритмы и структуры данных – М.; МИР, 1989. – 360 с.
2. Карпов Б.И. Delphi: Специальный справочник. – СПб: Питер, 2001-648с.
3. Карпов Б.И. Visual Basic Специальный справочник. – СПб: Питер, 2000-415с.WWW.codecrojekt.ru
4. WWW. master.ru
5. WWW.bdn_borland.com
6. http://microsofft.com

Маъруза №4.

Мавзу: Формал тил (ФЯ) ва формал грамматика (ФГ) тушунчаси

Режа:

1.Тил таърифи. Синтаксис ва семантика

2. Бэкус-Наур формасидаги грамматиканинг ёзилиши.

3.Формал тил (ФЯ) ва формал грамматика (ФГ) тушунчаси.

4. Белгилар занжири ва улар устидаги амаллар.

Калит сузлар.

- **Алфавит**
- **Белгилар занжири**
- **Лексика**
- **Синтаксис**
- **Семантика**
- **Грамматика**

1.Тил таърифи. Синтаксис ва семантика

Компиляторни ташкил этишдан аввал киритилаётган тилнинг аник таърифига эга булиш керак.

Бир неча каторлардан таркиб топган тилни тассаввур килишимиз мумкин. Тилни ифодалашда кандай каторлар ушбу тилга тегишли эканлиги (тил синтаксиси) ва ушбу каторларни киймати (тил семантикаси) аникланади. **Синтаксис** - формал тугри гаплар тупламининг коидалари тупламидир. Тилга тегишли каторларни тилнинг гаплари деб аталади. Реал тилларда чексиз гаплар сони булади ва уларни санаб утишнинг иложи йук. Энг содда тилнинг синтаксисини табиий тилда куйидагича ифодалаш мумкин, масалан: «барча каторлар, факат 1 ва 0 лардан ташкил топган» у холда 1111 ва 1000110 –тилга тегишли, 1020 эса йук.

Масалан, куйидаги гап «машина юради». «Машина» сузи эга, «юради» кесим. Ушбу гап куйидаги синтаксис коидалар ёрдамида ифодалаш мумкин булган тилга тегишли:

<гап>::=<эга><кесим>

<эга>::=машина | от

<кесим>::= юради | чопади

Ушбу учта каторнинг маъноси куйидагича: гап эга ва кесимдан иборат. Эга ёки машина деган бир суздан ёки от деган суздан ташкил топган. Кесим хам ёки юради деган суздан, ёки чопади деган суздан ташкил топган.

Ихтиёрий гапни бошлангич белги оркали кетма-кет куйиш йули билан олиш мумкин.

Ушбу коидаларни ёзишда фойдаланиладиган нотация **Бэкус-Наур формаси** деб аталади. Синтаксис бирликлар <гап> <эга> ва <кесим> нотерминал белгилар деб аталади, “машина”, ”от”, ”юради”, чопади терминал белгилар деб аталади, коидалар эса **тугилувчи коидалардир.** **::=, |**

. <> белгилар метабелгилардир. **Семантика** тилнинг барча гапларига киймат беради.

Алфавит –белгилар туплами. Масалан: Рус харфлари. Лотин харфлари , ракамлар.

Агар А-алфавит булса, А* А га кирувчи барча белгилардан тузилган каторларнинг (буш каторни хам кушган холда) тупламини англатади. А+ эса А га кирувчи барча белгилардан тузилган каторларнинг (буш каторни хам кушмаган холда) тупламини англатади. Буш катор купинча Е(эпсилон) ёрдамида белгиланади

Тилни синтаксисини тупламларни тасвирлаш оркали аниклаш мумкин, масалан L={0n1n|n>=0}. Ушбу тил бир ёки бир неча нуллардан, бирлардан ва буш катордан ташкил топган каторларни уз ичига олади.

Тилни мураккаброк синтаксисини грамматика ёрдамида аниклаш яхширок. Грамматикага тилни гапларини тузиш учун коидалар туплами киради. L синтаксисни оламиз ва куйидаги коидалардан фойдаланамиз.

1. S --> 0S1 2.S --> E

Ушбу тилнинг гапларини чикариш учун куйидагича иш юритамиз. S белгидан бошлаймиз ва уни 0S1 билан алмаштирамиз ёки Е билан. Агар S яна олинган каторда мавжуд булса, яна алмаштирамиз ва х.к. Шундай усул билан олинган S га эга булмаган катор шу тилнинг гапи хисобланади. Масалан, S 0S1 00S11 000S111 000111

Бундай каторларнинг кетма-кетлиги 000111 каторни чикиши дейилади, стрелка белгиси эса чикиш кадамларини булаклаш учун хизмат килади. Ушбу тилнинг барча гапларини иккита коидадан келиб чиккан холда келтириб чикариш мумкин, Ихтиёрий келтириб чикариш мумкин булмаган катор ушбу тилнинг гапи хисобланмайди. Грамматикани купинча кайта ёзиш тизими деб хам атайдилар.

Грамматика (Vt,Vn,P,S) туртлик билан аникланади, Бу ерда Vt – алфавит булиб, унинг белгилари терминаллар деб аталади, улардан грамматика оркали келтирилувчи занжирлар курилади. Vn –алфавит булиб, унинг белгилари нотерминаллар деб аталади, занжирларни куришда фойдаланилади. Vt ва Vn умумий белгиларга эга эмаслар, яъни Vt Vn =0, Грамматиканинг тулик алфавити V Vt U Vn каби аникланади.

P – тугилувчи коидалар туплами булиб, унинг хар бир элементи (a, b) жуфтлигидан ташкил топади, бу ерда a V+ да, ва b эса V* да.

a коиданинг чап булаги, b эса унг булагидир. Коида куйидагича ёзилади: a b. S Vn га тегишли ва **бошлангич белги (аксиома)**деб аталади. Бу белги тилнинг ихтиёрий гапини олиш учун таянч нуктадир.

L={0n1n|n>=0}.тилни генерация киладиган грамматика булиб

G0 =({0,1},{S},P,S), ,бу ерда P={S 0S1.S E} хисобланади.

L={anbm|n,m>=0}.тилни генерация киладиган грамматика булиб

G0 =({a,b},{S,A,B},P,S), бу ерда P={S AB, A aA, A E, B bB, B E} хисобланади.

S белгидан бошлаб нотерминални алмаштириш коидасини куллаб aaabb каторни генерация килиш мумкин.

S --> AB --> aAB --> aaAB --> aaaAB --> aaaB --> aaabB--> aaabbB --> aaabb

Хар бир бошлангич белгидан келтириб чикариладиган катор сентенциал форма деб аталади. Сентенциал формали гап –бу факат терминаллардан иборатдир. Терминалларни кичкина харфлар билан, нотерминалларни катта харфлар билан белгилаймиз.

Иккита бир хил тилни келтириб чикарадиган грамматикани **эквивалент грамматика** деб атаймиз.

2.Бэкус-Наур формасидаги грамматиканинг ёзилиши.

Куйидаги ёзув куриниши:

1)$\alpha \rightarrow \beta_1, \alpha \rightarrow \beta_2, \ldots, \alpha \rightarrow \beta_n$

$\alpha \rightarrow \beta_1 | \beta_2 | \ldots | \beta_n$

2)барча нотерминал белгилар <A> бурчак кавсларга олинади.

метабелгилардан фойдаланиб грамматика коидаларини ёзиш.

1.() – барча санаб утилганлардан факат биттасигина туриши мумкин.

2.[] – ушбу курсатилган занжирларлардан учраши хам мумкин учрамаслиги хам мумкин.

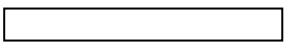
3.{} – ушбу кавсларда келтирилганлар учрамасликлари хам мумкин, ёки 1 марта учрашлари, ёки бир неча марта учрашлари мумкин.

4. , - () кавс ичидаги белгилар занжирини ажратиш учун фойдаланилади.

5. “ ” качонки метабелгилардан бирини занжирга оддий усул билан кушиш керак булганда фойдаланилади.

Грамматика коидаларини граф куринишида ёзиш.

Хар бир нотерминал белгига йуналтирилган граф куринишидаги диаграмма мос келади.

Номланиши	Белгиланиши	Йуналтирилиши
Кириш нуктаси	Хеч кандай белгиланмайди	Ундан графнинг кирувчи кобиги бошланади.
Нотерм белги		Ичида нотерминал белгиларнинг белгиланиши келтирилган
Терминал белгилар занжири		Унда терминал белгилар занжири ёзилган
Боглаш нуктаси		Чорраха
Кириш нуктаси	Хеч кандай белгиланмайди	Унга графни чикувчи кобиги киради

3.Формал тил (ФЯ) ва формал грамматика (ФГ) тушунчаси

Формал тил – бу сузлардан ташкил топган гапларнинг тупламидир (каторлар).

Катор – бу чекланган узунликдаги белгилар кетма-кетлигидир (белгилар бири биридан кейин ёзилган). Ушбу белгиларнинг хар бири аввалдан берилган алфавитнинг булаги хисобланади.

Алфавит – белгиларнинг ёки литераларнинг буш булмаган тугалланган туплами булиб, улар ёрдамида каторларни куриш мумкин. Формал тилда каторлардан гаплар курилади ва бу каторлар белгилар занжири деб аталади.

Изох:

Формал тилни берилиши учун унинг алфавит ива формал граматикасини курсатиш зарур.

Формал грамматика – каторлар тупламини ифодалаш учун керак буладиган коидалар тизимидир (белгиларнинг тугалланган кетма-кетлиги).

Уз навбатида каторлар гапларни ташкил этади ва х.к. Бундай каторлар туплами **тилни** ташкил этади. Тил кандайдир формал грамматика билан ифодаланади деб хисобласак, грамматика тилни юзага келтиради.

Фараз килайлик алфавит бу ={0, 1} булсин. Ушбу алфавитдан 01001 катор ташкил этилсин. Занжирдаги белгилар сони занжирнинг узунлиги m деб аталади (белгиларнинг мос тушишига богликсиз равишда). Буш занжир деб узунлиги m=0 булган занжир аталади. У .. ёки .. белгиланади.

Белги алфавитда аникланган каторлар тупламини англатади.

..={0,1,01,10,101,110,..}.

Буш туплам ихтиёрий тупламга киради.

L алфавитнинг формал тили .. дебкандайдир ихтиёрий.. . кисмтупламга айтилади. Знак .. разница с .. в одном символе. .. не включает в сея пустую строку.

Хар бир формал тил гаплари кандайдир коидалардан келиб чиккан холда тузилади. Бу коидалар **тилнинг синтаксисини** ташкил этади. Формал тилнинг муаммоларидан бири тил синтаксисини ифодалашдан иборатдир. Мазмуни: агар тиллар куп булмаган тугалланган гаплардан ташкил топган булганларида эди, у холда уларни санаб чикиш мумкин булар эди ва конструкцияларнинг тугрилиги ущбу тупламга тегишли ёки тегишли эмаслиги билан текширилар эди.

Лекин мумкин булган гаплар сони шунчалик купки, уларни санаб чикишнинг сира иложи йук.

Шунинг учун тиллар синтаксисини ифодаловчи махсус –**метатиллар** (тиллар устида тиллар) мавжуд. Ушбу тилда тил конструкциясини тугрилигини аникловчи коидалар тизимини ифодаланади.

Грамматика – тил синтаксисини коидалари тупламидир.

Грамматика икки хил куринишда булиши мумкин:

- Тугилувчи ;
- Англовчи .

Тугилувчи грамматика тугри гаплар ташкил этишни процедурасини бошлангич белигидан бошлаб ифодалайди.

Англовчи грамматика эса аник конструкцияни аник тилга тегишли эканлигини англаш жараёнини ифодалайди.(занжирдан бошлангич белгигача).

Изох:

Бу грамматикалар харакат йуналиши жихатидан фаркланадилар.

Тугилувчи грамматикага мисол куриб чикамиз.

Бу процедура рус тилини кесилган булаги куринишини эслатади. Грамматика объектлари: гап аъзолари, гап булаклари.

Белгиланиши:

ПР- предложение -гап

П – подлежащее- эга

С – сказуемое - кесим

ИС – имя существительное – от исми

М – местоимение - олмош

ГФ – глагольная форма – глагол куриниши

Коидалар:

1. <ПР>--<П><С>
2. <П>--<ИС>
3. <П>--<М>
4. <С>--<ГФ>
5. <ИС>--самолет
6. <ИС>--дом
7. <М>--он
8. <ГФ>--стоит
9. <ГФ>--строится
10. <ГФ>--летит

Белгиланиши:

a—b a b ни келтириб чикаради; a дан b чикади; b a дан келиб чикади.

Курсаткич чикиш йуналишини курсатади.

-- бор, кандай аникланади.

Изох:

Унгдаги нарса чапдаги нарсани аниклайди.

<имя> - нетерминал белги

имя – терминал белги.

Терминал белгилар грамматика коидалари ёрдамида очиб берилмайди, нетерминал белгилар эса очилади.

Грамматиканинг коидалар туплами **P** деб аталади.

Ушбу холда **P** санаб утиш куринишида келтирилган**.**

Ёки чап ёки унг кисмга кирувчи, ёки иккала кисмга хам кирувчи барча белгилар туплами **V** билан белгиланади.

P коидада **V** алфавитнинг белгиларидан тилнинг тугри гапларини тузиш хакидаги маълумотлар сакланади. (ушбу коидалар буйича тузилмаган гаплар нотугри хисобланадилар). Ушбу холатда $\mathbf{G_0}$ грамматика учун алфавит куйидагича белгиланади:

V = {<ПР>, <С>, <ИС>, <П>, <М>, <ГФ>, самолет, дом, он, стоит, строится, летит}.

Умумий холатда **V** туплам икки кисмдан **N** ва **T**, белгилар тупламидан тузилади.

Ихтиёрий нетерминал белги жуда булмаганда бир маротаба коиданинг чап томонига кириши шарт (**N** туплам).
N = {<ПР>, <П>, <С>, <М>, <ИС>, <ГФ>}
Т – факат унг булакка кирадиган терминал белгилар туплами.
Т = {самолет, дом, он, строится, стоит, летит}
Тугилувчи грамматики учун **S** – бошлангич белги.
S = {<ПР>}.
Компилятор учун (Паскаль) S сифатида «дастур» тушунчаси туради.
<Программа>--<Раздел описаний><Раздел действий>.

Агар унг булакда бир конструкция кейингисидан кейин катъий келса бу хол каторлар конкатенациясини англатади (занжирланиш) .

Ихтиёрий грамматика иккита масалани хал килиши керак:

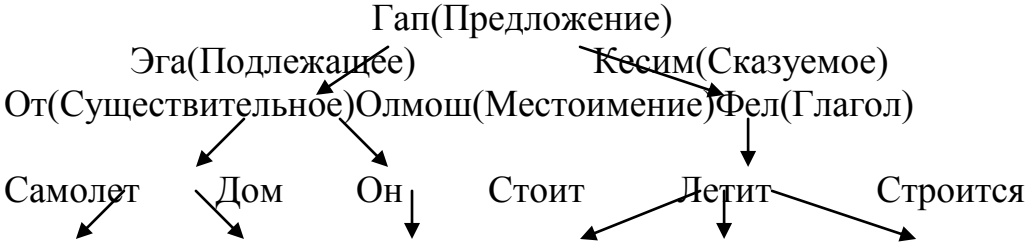
- Англаш масаласини (либо задачи распознавания);
- Ёки тугилиш масаласини (либо задачи порождения).

Тугилиш жараёнида тугри гапларнинг чикиши ифодаланади. Бу шундай амалга оширилади: бошлангич белгидан бошлаб чап булакнинг коидаларини унг булакка алмаштириш амалга оширилади.

Хар бир олинган тушунча узининг тарифига алмаштирилади. Бу жараён унг томонда факат терминал белгиларнинг узи колмагунича давом этади.

Жараённи грамматик разбор дарахти куринишида ифодалаш кулай. Ушбу дарахт кандай коидаларни кандай тил конструкцияларга куллаш мумкинлигини курсатади, лекин у аник тугилиш жараёнидаги куллаш тиртибини курсатмайди. Дарахт грамматиканинг ушбу коидаларига асосланган холда курилади. Юкорида бошлангич белги жойлашади, пастда – терминал белгилар. Дарахт №1 коидаларни куллаш йули билан курилади.

Гап эга ва кесимнинг конкатенациясидан ташкил топади. Кесим булиб ёки ИС, ёки М келиши мумкин.



Аник бир гапни чикиши ечим кабул килишни талаб этади: кайси йул билан пастга караб юриш керак. Дарахт эса коидаларни ифодалайди. Самолет строится – масалан.

Англаш масаласи дарахтдан фойдаланиб ечилади.

Дарахт буйича пастдан юкорига харакат килиб аник гапга бошлангич белгига етиб бориш керак. Бу ерда унг томон булакларини чап томон булаклари коидасига алмаштирилади.

Масалан, Дом летит.

ИС ва ГФ; --П ва С; --ПР.

Умумий холда грамматика :

1. Нетерминал белгилар туплами.

2. Теминал белгилар туплами.
3. Бошлангич белги.
4. Коидалар тупламидан ташкил топади.

$G = \{N, T, S, P\}$

Коидалар куйидаги куринишга эга: $a \rightarrow b$

$a \in (N \cup T)^+$

$b \supset (N \cup T)^*$

+ бу тупламга буш туплам киритиш мумкин эмаслигини англатади

* бу тупламга буш туплам киритиш мумкинлигини англатади

a ва b – баъзи бир каторлар (белгилар кетма-кетлиги)

Бундай коидалар **продукциялар** деб хам аталади.

G1 грамматикага мисол караб чикамиз. Бу холда куйидагилар киритилади:

Нетерминал белгилар $\rightarrow$ A, B, C, …

терминал белгилар $\rightarrow$ a, b, c, …

Караймиз

$G1 = \{N, T, S, P\}$, бу ерда

$N = \{A, B, S\}$

$T = \{a, b\}$

$P = \{ S \rightarrow AB$ (1)

$A \rightarrow aA$ (2)

$A \rightarrow a$ (3)

$B \rightarrow Bb$ (4)

$B \rightarrow b$ (5)

}

№2 коидада нетерминал белги А хам чап хам унг булакда мавжуд. Бу эса А белги тегишли каторлар синфи А белгига а префиксни кушиш оркали курилишини англатади.

Учинчи коидада А а оркали аникланади. Умумий холда бундай жараён каторлар конкатенацияси деб аталади (А катор чапдан кушиладиган а конкатенациясидан курилади).

Кандайдир тушунча узи узидан куриладиган холат рекурсия деб аталади.

Туртинчи коидада хам рекурсия мавжуд. Занжир белгини унгдан кушиш йули билан ташкил этилади.

Бу холатда коидалар барча мумкин булган вариантларни санаб чикиш йули билан берилади, хар бир вариант учун бита катор.

Грамматика коидаларини бериш усули нотация деб аталади.

Купинча Бэкус Наура формасидан фойдаланилади.Унда куйидаги белгилашлардан фойдаланилади: $\rightarrow$; ::=

Барча нетерминал белгилар бурчак кавсларга олинади. Агар кандайдир тушунча учун чап булакда бир нечта вариант бор булса, у холда “|” белгидан фойдаланилади.

<A> ::= a | a<A>

Бундай ташкари грамматика коидалари метабелгилардан фойдаланилган холда хам берилади (кавслар):

( ) – думалок кавсларда санаб утилган барча конструкциялардан ушбу конструкция учраган вактида факат биттагинасидан фойдаланилади.Вергулдан булаклаш учун фойдаланилади.

[ ] – бу кавсларга киритилганлар булиш хам, булмасликлари хам мумкин.

{ } – такрорлашни англатади (n марта, n = 0,1,2,…. 0 конструкция мавжуд эмаслигини англатади.)

Метабелгилик ва метабелгисиз ишоралик бутун сонлар учун коидаларни берилишини караб чикамиз.

G = {{0,1,2,…,9},{<ишорали сон>,<сон>,<ракам>}, P,<ишорали сон>}

P = {

<ишорали сон> ::= +<сон> | -<сон>

<сон> ::= <ракам> | <ракам><сон>

<ракам> ::= 0|1|2|3|4|5|6|7|8|9

}

Иккинчи коида рекурсив.

Худди шунингдек метабелгилик ишоралик сонлардан фойдаланилган хол учун :

< ишорали сон > ::= [(+,-)] ракам{ракам}

4.Белгилар занжири ва улар устидаги амаллар.

Белгилар занжири – бу бири биридан кейин ёзилган белгиларнинг ихтиёрий кетма-кетлигидир.

Белгилар занжири (БЗ) учун таркиб, белгилар сони ва тартиб мухимдир. БЗ α ва β тенг α=β ёки мос тушадилар, агар улар битта белгилар таркибига эга булсалар, бир хил белгилар сонига ва белгиларнинг занжир буйлаб бир хил келиш тартибига эга булсалар. Занжирдаги белгилар сони занжир узунлиги дейилади.

БЗ куйидаги хусусиятларга эгадирлар:

1)Конкатенация – 2 та занжирни йигиндиси ёки купайтмаси αβ

α=“BA”

β=“CL” => αβ=“BACL”

Конкатенация амали коммутиция хусусиятига эга эмас, яъни $\alpha\beta \neq \beta\alpha$. Ассоциативлик хусусиятига эгадир $(\alpha\beta)\gamma=\alpha(\beta\gamma)$

2)Занжирга мурожат – занжир белгиларини тескари тартибда ёзиш α^R, α=“ВАСЯ” => α^R=“ЯСАВ” Ушбу амал учун $(\alpha\beta)^R=\alpha^R\beta^R$ хакикат

3)якинлашув – занжирни n марта такрорлаш

4)белгиларнинг буш занжири – бу битта хам белгига эга булмаган занжирдир, λ -буш занжир учун куйидаги хакикат: 1) |λ|=0; 2) ихтиёрий α: $\lambda\alpha=\alpha\lambda=\alpha$; 3)$\lambda^R=\lambda$; 4) ихтиёрий $n\geq 0$:$\lambda^n=\lambda$; 5)ихтиёрий α: $\alpha^0=\lambda$

Назорат саволлари

1. Тил синтаксисини нималар аниклайди?
2. Тилни синтаксиси ва семантикаси орасида кандай фарк бор?
3. Грамматика нима ва у кандай берилади?
4. Тилнинг терминал ва нотерминал белгилари кандай фаркланадилар?
5. У ёки бу белгининг тилнинг гапларида учрашини кандай изохлайсиз?
6. «Бошлангич белги» нима ва у тилнинг бошка белгиларидан нима билан фарк килади?
7. Грамматика тугилувчи коидалар оркали берилган булсин.

   S ->(S) S->E,
   S->SS бу ерда S – бошлангич белгиб, E буш катордир.

   Куйидаги каторлар ушбу грамматика буйича генерация килинган тилга тегишлими? Жавобингизни исботланг.

а) катор((1)()), в) катор (()()())

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза 5.

Мавзу:Трансляторларни қуришнинг асосий тамойиллари.

Режа:

1. **Компиляция назариясининг элементлари.**
2. **Компиляторнинг ишлашининг умумий чизмаси.**
3. **Компиляторнинг структураси. Трансляция килинадиган дастурларнинг турлари.**

Калит сузлар.

- **Компилятор**
- **Объект дастури**
- **Алфавит**
- **Лексика**
- **Лексема**
- **Синтаксис**

1.Компиляция назариясининг элементлари

Компилятор – тизимли кайта ишловчи дастур булиб, юкори даража дастурлаш тилида ёзилган дастурни машина куринишига угиради (объект дастурга).

Объект дастури машина кодларининг кетма-келигидан иборатдир.

Компиляторнинг иши алфавитни, лексикани, синтаксисни семантикани (мазмунни) саклаган холда узгартиришдир.

Алфавит – тил конструкциясини куришда фойдаланиладиган мумкин булган белгилар тупламидир.

Лексика – тилнинг алохида маъноли бирликлари тупламидир (лексемалар).

Лексема – тилнинг маънога эга булган минимал конструкциясидир.

Лексемаларга узгарувчилар исмлари, хизматчи сузлар, константалар, амал белгилари, ажраткичлар ва х.к.

Синтаксис – тилнинг тугри ифодалаш имконини берадиган коидалар тизимидир.

Семантика – синтаксис жихатидан тугри конструкцияларнинг маъновий булагини ифодалайди.

Алфавит, лексика, синтаксис формал куринишда берилиши мумкин. Семантика эса купинча формаллашган эмас, семантика коидаларининг текшириш алгоритмлари куринишида берилади.

2.**Компилятор ишининг умумий чизмаси икки фазадан ташкил топади:**

1. тахлил фазаси

2. синтез фазаси

1. Тахлил фазаси лексик анализатор ишидан бошланади. (сканердан).

У киритилган дастурнинг караб чикишдан бошланади, турли лексемларни ажратиш ва классификацияси ва кейинги боскичга узатиладиган

мос жадвалларни тулдиришдан иборат. Сканер куйидаги жадваллардан фойдаланади:

1) Тилнинг терминаль белгилари жадвали.

У сканер иши учун бошлангич хисобланади ва компиляторни ишлаб чикарувчиси томонидан курилади.

Терминаль белгилар куйидагилардир:

- ажраткичлар
- амал ишоралари
- тилнинг калит сузлари

2) Белгили исмлар жадвали (идентификаторлар).

Сканер бу жадвалга хар бир учраган исмни киритади, агар у коидага асосан курилган ва калит сузларга тегишли булмаса.

3) Константалар жадвали.

Хар бир константа учун куйидаги характеристикалар тегишлидир:

- Киймат
- Тур
- Асоси (Основание СС)
- Кенглик (хотира сохаси улчами)

Натижада дастур жадваллар тупламига айланади. Бундай шакл компилятор ишининг куйидаги булаги учун кулай хисобланади.

Бундан ташкари, хар бирида 2 тадан майдони булган ёзувлардан ташкил топган лексемлар жадвали курилади.

Лексем туридаги код	Жадвалдаги ракам
I	5

I – лексема идентификатор эканлигини белгисидир.

5 – идентификаторлар жадвалидаги урин.

Хар бир лексемага лексемалар кодлари жадвалидаги уз ёзуви мос келади, бу ерда биринчи ёзувга биринчи лексема мос келади.

Тахлил фазасининг кейинги булаги – синтаксис анализатордир (parser).

Бу булакда компилятор тилнинг барча конструкцияларини тулик синтаксис текширувини бажаради.

Конструкция – маънога эга булган лексемаларниг кандайдир кетма-кетлигидир.

Бу булак констукцияларни ажратиш ва тугрилигини аниклашни бажаради (лексемларни кетма-кетлиги ва мажбурий ва мажбурий булмаган тушунчаларни борлигини). Бу булак мос ёзувлар килинадиган хатоликларни топади. Хатоликларни топилгандан сунг иш ёки давом эттирилади, ёки тухтатилади.

Синтаксик анализатор ишининг кириш маълумотлари булиб лексем кодлари жадвали хисобланади.

Кейинги булак – семантик анализатор. Конструкцияларнинг тугрилиги урнатилгандан сунг, семантик анализатор ишга туширилади. Бу ерда

конструкция маъносининг интерпретациясини текширилади. Купинча синтаксис тугри булган конструкция семантика нуктаи назаридан нотугри булиши хам мумкин. Масалан:

```
Var
  a: integer;
Const
  N:=2;
Begin
  N:=a;
```

Семантика нуктаи назаридан нотугри, чунки константага киймат узлаштириш мумкин эмас.

Семантик анализатор хар бир синтаксис конструкция учун кандайдир кийматни мос куяди, улар эса куйидаги формалардан бири булиши мумкин:

- Машина кодлари формаси
- Ассемблер тилининг формаси
- Оралик форма

Купинча оралик формадан фойдаланилади:

1) Тетрад формаси. Бу ерда хар бир операторга тетрада мос куйилади. Тетрада куйидаги куринишга эга: **T1,T2,T3,T4**

T1 – амал

T2,3 – операндлар

T4 – натижа

Масалан: A:=B + C*Д; $\rightarrow$*,C,Д,T1 +,B,T1,T2 :=,T2,_ ,A

Бу ерда T1,T2 – оралик узгарувчилар.

2) Польяк ёзуви:

A:=B + C*Д $\rightarrow$ СД*B + A=

Семантик анализатор иши билан тахлил фазаси тугайди.

2. Синтез фазаси дастур узгарувчилари буйича хотирани таркатишдан бошланади.

Бу ерда белгили исмлар жадвали каралади, ва хар бир исм учун хотира ажратилади, яъни бошлангич манзил ва соха улчови белгиланади. Шундай килиб шу йул билан оралик узгарувчиларга хотира ажратилади.

Хотиранинг бундай ажратилиши **хотиранинг статик ажратилиши** деб аталади.

Статик узгарувчилардан ташкари динамик узгарувчилар хам мавжуд. Уларга хотира куйидаги куринишда ажратилади: тескари кодга дастурнинг бажарилиш жараёнида бажариладиган коднинг франментлари куйилади. Натижада хотирани таксимлаш жадвали ташкил этилади.

Иккинчи дастур – машина командалари генерациясидир. Ушбу боскичда дастурни ички ифодалашлардан машина кодларига утказиш амалга оширилади. Киритилувчи берилганлар сифатида амалларни кодлари жадвалидан (хар бир процессор учун узиники) фойдаланилади. Оралик амалларнинг хар бир тури учун бир ёки бир неча машина командалари тузилади. Бу машина командаларига операндларнинг кийматлари ёки манзиллари куйилади ва тулдирилган майдонли машина кодлари олинади.

Кейинги боскич – кодни оптималлаш. Купинча бу жараён икки йуналишда амалга оширилади:

- Дастур хажмининг кискариши
- Ишнинг тезлаштирилиши

Икки хил услуб мавжуд:

- Машина командалари даражасида оптималлаш
- Оралик кодлари даражасида оптималлаш, - купинча фойдаланилади.

Бу схема ишида синтаксис ва семантик тахлил аник булинади. Бундай иш схемаси кенг фойдаланувчилар оммасига мулжалланган ва синтаксис усуллардан фойдаланадиган компиляторларни куришни англатади.

Бу схемадан ташкари компиляциянинг тугри усуллари хам кулланилади, уларда семантика ва синтаксис тахлили аник ажратилмайди ва 1 кирувчи тилга мулжалланган.

Синтаксис усуллар умумий хисобланади ва кенг кулланилади. Синтаксис усуллар формал тиллар назариясига ва формал граматикага асосланиб курилган.

3.Компиляторнинг структураси. Трансляция килинадиган дастурларнинг турлари.

Хар бир компилятор оддий командаларнинг чекланган тупламини бажариш кобилиятига эга. Ихтиёрий мураккаб харакатлар ушбу командаларнинг кетма-кетлиги куринишида тасвирланади. Юкори даража дастурлаш тилида ёзилган дастурни бажариш учун, уни авваломбор машина кодларидаги командалар кетма-кетлигига утказилади.

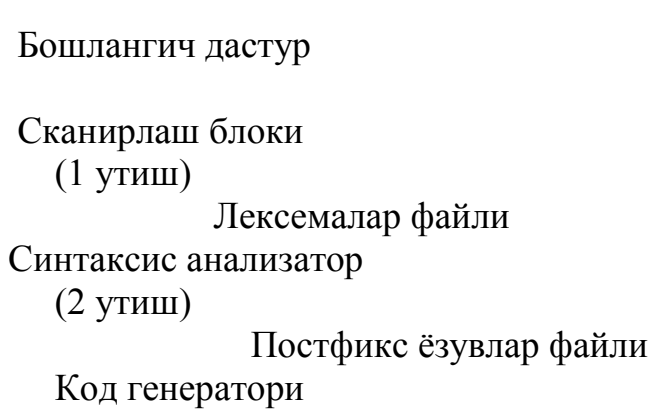
Бошлангич берилган дастур (кандайдир дастурлаш тилида ёзилган) белгилар кетма-кетлигидан иборат булиб, улар компьютерга киритилади ва бажарилиш учун керак булган куринишга айлантирилади.

Компилятор бу шундай, аник бир куринишдаги белгилар каторини (яъни берилган дастурлаш тилидаги дастур матнини) кабул киладиган ва бошка белгилар каторини (машина тилидаги дастурни) чикарадиган дастурдир. Компиляторларга бир катор умумий хусусиятлар хоски, улар компиляция килинаётган дастурларни ташкил этиш жараёнини соддалаштирадилар. Ихтиёрий компилятор таркибига куйидаги учта асосий компонента киради:

- лексик анализатор (сканирлаш блоки);
-синтаксис анализатор;
-машина командалари кодлари генератори.

Анализаторларнинг харакат тамойилларини формал моделлар ёрдамида ифодалаш мумкин булган холда, кодлар генератори учун умумий формал тасаввурлар мавжуд эмас. Лексик анализ фазасила дастурнинг бошлангич берилган матни лексемалар деб аталган бир бири билан богланмаган занжирлар куринишидаги белгиларга (бирликларга) ажратилади. Бундай матнли бирликлар калит сузлар (IF, DO ва бошкалар), узгарувчилар исмлари, константалар ва амалларнинг ишоралари (+,- ёки *). Бундай сунг бу сузлар алохида белгилар гурухи эмас, булинмаслар сифатида каралади. Дастурни

лексемаларга булаклангандан сунг, грамматик разбор деб аталувчи ва операторларнинг кетма-кетлигининг тугрилигини текширадиган, синтаксис анализ фазаси келади. Мисол учун, «IF ифода THEN гап» куринишига эга IF гапи учун грамматик разбор куйидагича: IF лексемасидан кейин тугри ифода келади, ушбу ифодадан кейин THEN лексемаси келади, ундан сунг яна тугри ифода келади ва у «;» белгиси билан тугайди. Охирида кодни генерациялаш жараёни бажарилади ва синтаксис анализ натижаларидан фойдаланилиб бажаришга тайёр машина тилидаги дастур ташкил этилади. Ихтиёрий компилятор таркибига юкорида айтиб утилган учта компонента кирса хам, уларнинг узаро харакати турли усуллар оркали амалга оширилади. Ушбу компоненталар орасидаги узаро харакатларнинг кенг таркалган вариантларини караб чикамиз.



1- расм. Уч утишли транслятор.

Сканерлаш блоки бошлангич дастурни укийди ва лексемалар файли сифатида ифодалайди. Синтаксис анализатор эса бу файлни укийди ва дастурни янги ифодасини чикаради, масалан постфикс куринишида. Ва нихоят, бу файл код генератори оркали укилади ва дастурни объект коди ташкил этилади.

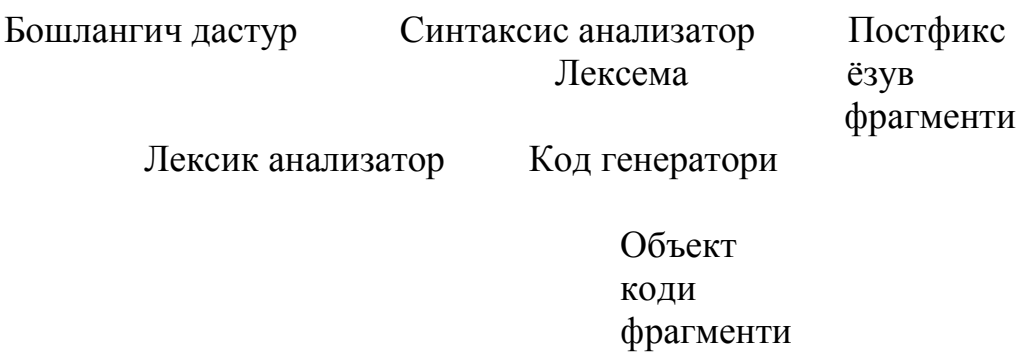
Бундай куринишдаги компилятор уч утишли компилятор деб аталади, чунки дастур бу жараёнда уч марта укилади (бошлангич дастур матни, лексемалар файл ва постфикс куринишидаги файл).

Камчиликлари: Бажарилишнинг жуда юкори булмаган тезлиги, чунки купгина операцион тизимларда файлларга мурожаат амаллари нисбатан секин бажарилади.

Ютуклари: Компияциялаш жараёнининг хар бир фазасининг нисбатан бири –бирига боглик эмаслиги. Чунки кайта ишланаётган блоклар уртасидаги алока факат берилганларнинг файллари оркали амалга оширилади, ихтиёрий утиш колганларига боглик булмаган холда амалга оширилиши мумкин. Бу эса:

1. Компиляторнинг турли блокларини турли ишлаб чикарувчилар автоном равишда ишлашлари имкониятини яратади ва факатгина оралик файлларни формаларини мослаштириш зарур булади.
2. Компиляторнинг эгилувчанлиги. Масалан, турли турдаги компьютерлар учун бир тилдан фойдаланиш, сканерлаш ва синтаксис анализаторларнинг бир хил блокларидан фойдаланиш, лекин хар бир тур компьютер учун махсус код генераторларини ёзишни англатади. Бир турдаги компьютерлар учун турли тиллардан фойдаланишда сканерлаш ва синтаксис анализаторларнинг турли блоклари талаб килинади, лекин улар учун умумий код генераторидан фойдаланиш мумкин.
3. Оператив хотира хажмига минимал талаблар (компиляциянинг турли фаза модулларини навбатма-навбат, аввалгисини чикариб ташлаган холда, оператив хотирага юклаш).

Компиляциянинг юкори тезлигига эришиш учун бир утишли структурага эга компилятор кулланилади. Суратда бошкарув буйича алокалар узлуксиз чизиклар билан, берилганларни узатиш пунктир чизиклар билан курсатилган.



2- сурат. Бир утишли транслятор.

Бу холда синтаксис анализатор асосий бошкарувчи дастур сифатида кисм дастурлар куринишида ташкил этилган сканер блоки ва код генераторини чакиради. Синтаксис анализатор доимий равишда сканерлаш блокига каралаётган дастурдан бир лексемадан кейин бошка лексемани олган холда постфикс ёзувли янги элемент учрагунига кадар мурожаат этади, ундан кейин код генераторига ушбу дастур фрагменти учун объект кодини ташкил этувчи код генераторига мурожаат килади.

Ютуклари: Максимал унумдорлик ва бажарилиш тезлиги. Чунки дастур бир марта каралади ва файлга мурожат амаллари минимал булади (факат берилган файлдан укиш ва объект файлларига ёзиш).

Камчиликлари:

1.Олдинга утишни ташкил этишдаги муаммолар. Масалан: куйидаги гапни карасак,

GO TO метка;

«метка» хали дастур матнида учрамаганлиги учун баъзи кийинчиликлар юзага келиши мумкин.

2.Ташкил этилаётган объект дастурни оптимал эмаслиги. Масалан, агар куйидагича матн учраса:

A=(B+C);

P=(B+C)+(E+M); компилятор дастурни куйидаги куриншга келтириб, ундан фойдалирок объект кодини ташкил этиши мумкин эди.

A=(B+C);

P=A+(E+M);

Лекин бир утишли компилятор жуда куп керакли маълумотни матнда (E+M) формула учрагунига кадар йукотиши мумкин.

3. Бир утишли компилятор хотирада тулик жойлашганлиги сабабли, уни амалга ошириш хотира ресурсларига юкори талабларни куяди.

Объект дастур бажарилиши унумдорлигини ошириш учун компиляция жараёнига оптимизация фазасини кушиш мумкин. Оптимизация блоки уч утишли компиляторга синтаксис анализатор ва код генератори уртасига осон жойлаштирилади. Бу фазада постфикс файл кирувчи берилганлар сифатида фойдаланилади ва янгиланган характеристикали дастурга эквивалент постфикс ёзувли янги файл ташкил этилади. Оптимизация блоки узининг чикувчи натижаларини постфикс файл форматида ёзганлиги сабабли, кодлар генераторига уларни узгартириш шарт булмайди. Амалиётда оптималлаш имконияти фойдаланувчи хохишига кура амалга оширилади, агар компиляция вакти жуда куп вакт олмасин десак, у холда оптимизация блокини бажармаймиз, агар бажарилиш тезлиги юкори булган дастур керак булса, у холда синтаксис анализатор ишидан сунг оптимизация блоки чакирилади.

Юкорида куриб утилган икки вариантдан оралик холатни эгалловчи икки утишли компиляторни караб чикамиз. Бу холда синтаксис анализатор сканерлаш блокини чакиради, лексемаларни кетма-кет укийди ва дастурни постфикс ёзувлар файлини ташкил этади. Код генератори Ушбу файлни укийди ва дастурни объект кодини ташкил этади. Бундай структурага нисбатан кам бажарилиш вакти сарф булади, чунки дастур икки марта укилади (бошлангич мант ва постфикс ёзувлар файли). Бу холатда олдинга метка буйича утиш муаммоси хал этилади, чунки Ушбу метка биринчи утиш фазасида код генераторини чакиришдан аввал укилади. Бундай компиляторга зарурият булганда оптималлаш блокини кушиш осон амалга оширилади. Барча куриб утилган компиляторлар маълум иш шароитларида узининг ютук ва камчиликларига эга.

Интерпретаторлар худди компиляция га ухшаш тамойилни амалга оширадилар. Компиляторлар ва интерпретаторлар купгина умумий хусусиятларга эгалар. Интерпретатор хам авввалombор бошлангич дастурни караб чикади ва ундаги лексемаларни ажратади. Бунинг учун худди

компияция килувчи дастурлар таркибига кирувчи сканерлаш блоки ва систаксис анализаторлардан фойдаланилади. Лекин интерпретатор у ёки бу амалларни бажарадиган объект кодни куриш урнига узи мос харакатларни амалга оширади.

Интерпретаторни ютуклари:

- амалга оширишнинг нисбатан соддалиги;
- дастурларни кайта ишлашнинг кулайлиги.

Компиляторнинг ютуклари:

- бажарилиш тезлиги;
- бажарилувчи коднинг дастурлаш тизимларига боглик эмаслиги;
- буюртмачиларга дастурларни бошлангич матнсиз бериш имконияти.

С, С++, Паскаль, Delphi каби дастурлаш тилларида компиляторлардан фойдаланилади. Бейсик дастурлаш тилининг бир канча вариантлари ва СУБД ларнинг купчилиги интерпретаторлардан фойдаланилади. Тармокли дастурлаш тили Java махсус виртуал Java машина сифатида амалга оширилган.

Компилятор бошлангич дастурни битлар тупламига айлантирганлиги сабабли, ушбу битлардан бошка машинада хам фойдаланса булади. Бир турдаги машина учун дастурни тайёрлаш бошка бир компьютерда амалга оширилган холда, бунадй компиляторлар **кросс** компиляторлар деб аталади.

Конвертор - бир тилдан бошка уша даражадаги тилга угириш демакдир. Мисол сифатида Паскаль тилидаги кодни СИ тилидаги кодга айлантирувчи дастурни келтириш мумкин.

Транслятор деганда биз бир белгилар каторини (бошлангич матнни) бошка белгилар каторига (объект дастурга) айлантирадиган дастурни тушунамиз. Бу жараённинг натижаси булиб, у ёки бу машина учун машина тилидаги дастур, ёки бошка бир тилдаги бошлангич дастур матни булиши мумкин. **«Компиятор»** терминини бошлангич дастур матнини машина тилига угирадиган дастур сифатида караб фойдаланамиз. Агар трансляция жараёнининг барча куринишлари эътиборга олинса у холда **«транслятор»** терминини ишлатамиз.

Назорат саволлари.

1. Бир-икки ва уч утишли компиляторларнинг ютуклари нималарда куринади?
2. Компилятор ва интерпретаторнинг фарки нимада?
3. Конверторлар кайси холатларда кулланилади?
4. Кросс-дастурлар нима учун керак ?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с

3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза №6.

Мавзу: Грамматикалар классификацияси.

1. **Хомский иерархияси.**
2. **Грамматикалар классификацияси**
3. **Тиллар классификацияси.**
4. **.Контекст-озод грамматикалар.**

Калит сузлар.

- **Ноль грамматика**
- **контекст-боғлиқ грамматика**
- **Контекст – озод грамматика**
- **Регуляр грамматика**
- **Иккинчи даражали грамматика**
- **Регуляр ифода**
- **Норегуляр ифодага**
- **Фаза**
- **Уз-узига юклаш**

1.Хомский иерархияси.

Грамматикада учраши мумкин булган коидалар турларининг чегараланганлиги катор махсус грамматика синфларини аниклаш имкониятини беради. Улардан бир Хомский иерархиясидир. Уни куйидагича ифодалаш мумкин:

1. Аввалдан аникланган ихтиёрий грамматика – **0 турдаги грамматикадир.**
2. a → b курнишдаги барча коидалар учун белгилар сони мос равишда a ва b булса, у холда бу грамматика **1 турдаги грамматика** деб аталади ёки **контекст-боглик (КЗ) грамматика** деб аталади.
3. Грамматика коидаларининг барча чап булаклари биттадан нотерминал белгидан ташкил топса, у холда бу грамматика **2 турдаги грамматика ёки контекс-озод (КС) грамматика** деб аталади.
4. Грамматика **3 турдаги грамматика** ёки **регуляр грамматика** деб аталади, агар грамматиканинг хар бир коидаси куйидаги формалардан бирига эга булса:

Унг томонга тугриланган грамматика холида грамматиканинг унг кисмида биттадан куп булмаган нотерминал мавжуд булади ва у энг унг белги булади:

A → a

A → aB

Чап томонга тугриланган грамматика холида грамматиканинг унг кисмида биттадан куп булмаган нотерминал мавжуд булади ва у энг чап белги булади:

A → a
A → Ba

3 тур грамматикасининг барча куринишларини уз ичига олган шажара 2 тур грамматикаси ва х.к. хисобланади. Грамматика иерархиясига тиллар иерархияси мос келади. Масалан, 2 тур грамматикаси буйича тилни генерация килиш мумкин булса, у холда бу тилни 2 тур тили деб аталади. Шуни эсда тутиш керакки, тилни генерациялаш учун бир неча грамматикадан фойдаланиш мумкин булса, тилнинг тури энг куп турли грамматикага мос келади.

Хомский иерархияси турли тилли трансляторларни куриш нуктаи назаридан мухимдир. Грамматикада канчалик кам чегаралар мавжуд булса, генерация килинаётган тилга куйиладиган чегаралар шунчалик мураккабдир. Фойдаланилаётган грамматика синфи канчалик универсал булса, биз тилнинг хусусиятларини шунчалик купрок ифодалай оламиз. Лекин, грамматика канчалик универсаль булса, ушбу тилни англовчи дастур шунчалик мураккаб булади.

3 тур грамматикани дастурлаш тилларининг бир неча хусусиятларини ёки аппаратуранинг ифодалашнинг юкори даража тилларни ифодалаш учун фойдаланилади. Масалан, купгина дастурлаш тилларининг идентификаторларини генерациялаш учун куйидаги коидалардан фойдаланиш мумкин.

L → l	R → d
L → lR	R → lR
R → l	R → dR

Бу ерда (l) ва ракам (d) терминал белгиларни англатади.

Купинча коидаларнинг бир хил унг ва чап томонларини бирлаштириш кулай.

Юкорида келтирилган грамматикани куйидаги куринишда ёзиш мумкин:

L → l | lR

R → l | d | lR | dR

Вертикал чизик «ёки» маъносини англатади.

Дастурлаш тилларининг купгина «локал» курилмалари, масалан константалар, калит сузлар ва каторлар 3 тур грамматика ёрдамида ифодаланади. Аппаратуранинг жуда содда ифодалаш тилларини регуляр грамматика ёрдамида ифодалаш мумкин. Лекин 3 тур грамматикаси

факатгина катъи чекланган тилларнинг турларини яъни регуляр ифодаларни генерациялайди.

А алфавитда регуляр ифодаларга куйидагилар киради:

1. Элемент А (ёки буш катор).

Агар P ва Q регуляр ифодалар булса, у холда куйидагилар хам регуляр ифода хисобланадилар:

2. PQ (Q P дан кейин келади)

3. P | Q (P ёки Q)

4. P * (нул ёки P нинг куп экземплярлари)

Алфавитда {a,b,c} ab*|ca* - регуляр ифода ва у куйидаги каторларни уз ичига олувчи тилни ифодалайди: abb c caaa ab ca

Регуляр ифодага мисол.

Идентификаторни ифодаловчи регуляр ифода куйидаги куринишга эга: L(L\D)*, бу ерда L харфни англатади, D ракамни англатади.

Регуляр ифодаларда чегаралар мавжуд. Масалан, регуляр ифодалар ихтиёрий узунликдаги кавслар шаблонини бера олмайди ва мос равишда уларни 3 тур грамматикаси ёрдамида генерация килиб булмайди.

Норегуляр ифодага мисол.

Очилган ва ёпилган кавслардан ташкил топган тилни караймиз (буш каторни хам кушган холда) у куйидаги хусусиятларга эга:

1) Чапдан унгга укиш жараёнида учраган ёпилган кавслар сони хеч качон очилган кавслар сонидан ошиб кетмайди.
2) Хар бир каторда очилган ва ёпилган кавслар сони бир хил булади.

Масалан: куйидаги каторлар тилга тегишлидир.

( ) (( ) ( ) (( )))

(( ) ( ) (( ) (( )))) (( ))

Куйидагилар эса тегишли эмас:

((( ) ( )) 2 коидага мос эмас.

( ))) ((( ) 1 коидага мос эмас.

Ушбу тилнинг регуляр ифодалар ёки 3 тур грамматикаси ёрдамида уни генерациялаш усули мавжуд эмас. Лекин ушбу тил куйидаги озод –контекст грамматикаси ёрдамида курилади:

S → (S)
S → SS
S → e

Купгина дастурлаш тилларида ва аппаратурани ифодалаш тилларида шундай кавслар жуфтлиги борки, уларни келиши зарур, масалан:

( ) , [ ] , begin end хар бир очилган кавсга ёпилган кавс мос келиши керак.

Масалан begin ( ) end - тугри
begin (end) - нотугри.

Озод –контекст грамматика ушбу чегараларга йул куяди. Дастурлаш тиллари синтаксисининг ва махсус САПР тилларининг купгина кисми КС- ушбу грамматика оркали ифодаланади. Аммо купгина тилларда баъзи бир хусусиятлар борки, уларни Кс-грамматика оркали ифодалаб булмайди. Масалан: Х:=У мумкин булиши мумкин, агар Х ва У мос турга эга деб эълон килинган булса, ёки мумкин булмаслиги мумкин, агар турлар мос келмасалар. Бундай шартлар КС грамматика ёрдамида амалга ошириб булмайди, шунинг учун компиляторлар купинча текширувни формал синтаксис анализ фазасида бажара олмайдилар. Лекин КС грамматика гоясини, тилнинг баъзи бир контекс-боглик хусусиятларини кушган холда, кенгайтириш мумкин.

2.Грамматикалар классификацияси

Нол тур: Фраза структурали грамматика. Коидалар структурасига хеч кандай чегаралар куйилмайди: $\alpha\rightarrow\beta$, бу ерда $\alpha\epsilon V^{+}$, $\beta\epsilon V^{*}$ - жуфт барчаси грамматикани чикармаган холда. Амалиётда кулланилмайди, V=VTUVN, G(VN,VV,P,S).

Биринчи тур: контекст-боглик: а)кискартирувчи: $\alpha_1 A\alpha_2\rightarrow \alpha_1\beta\alpha_2$, бу ерда α_1 ва $\alpha_2 \epsilon V^{*}$, $A \epsilon VN$, $\beta \epsilon V^{+}$. G(VT,VV,P,S) б) кискартирилмайдиган грам: $\alpha\rightarrow\beta$, бу ерда α ва $\beta \epsilon V^{+}$, $|\beta|>=|\alpha|$. Контекс –боглик грамматика коидалари структураси ушбу тил гапларини куриш жараёнида битта нотерминал белги у ёки бу занжирга контексга боглик холда кушилган булиши мумкин ва унда у учрайди: $\alpha_1 A\alpha_2\rightarrow$(ёки α_1 , ёки α_2 нолга тенг). Кискартирилмайдиган грамматика: худди шундай структурага эга, лекин А ва β занжирлар узунлиги бир хил. Компиляторларни куриш жараёнида ушбу грамматикадан фойдаланилмайди.

Иккинчи тур: контекст-озод граматика: $A\rightarrow\beta$, бу ерда $A\epsilon VN$, $\beta\epsilon V^{+}$(кискартирилмайдиган КБ грам). Кискартириладиган КО: $A\rightarrow\beta$, бу ерда $A\epsilon VN$, $\beta\epsilon V^{*}$. Коиданинг унг кисмида буш белгилар занжири булиши мумкин.К-О грамматика дастурлаш тилларининг синтаксис конструкцияларини ифодалашда фойдаланилади.

Учинчи тур:Регуляр грамматика: чап чизикли : $A\rightarrow B\gamma$, $A\rightarrow\gamma$,бу ерда А ва В ϵVN, $\gamma \epsilon VT^{*}$, унг -чизикли: $A\rightarrow\gamma B$, $A\rightarrow\gamma$, бу ерда А ва В ϵVN, $\gamma \epsilon VT^{*}$. Регуляр грамматика дастурлаш тилларининг оддий конструкцияларини ифодалашда фойдаланилади. Компиляторда унинг асосида кирувчи тилнинг лексик тахлил функциялари курилади.

Энг мураккаб грамматика бу нол грамматикадир. Энг содда 3 тур грамматикасидир.

3.Тиллар классификацияси.

Тур 0: фраза структурали тиллар: 0 грамматика оркали берилган энг мураккаб тилдир. Чекланган захираларда, куп вакт давомида тилнинг

гапларини разборини бажара оладиган компиляторни куриш мумкин эмас.Табиий тил компиляторини куриш мумкин эмас.
Тур 1: контекст-боглик тиллар: 1 тур грамматикаси оркали берилади, тилнинг гапларини англаш вакти белгилар занжирининг узунлигига экспоненциал боглик. Матнни таабиий тилга тахлили ва угиришда фойдаланилади.
Тур 2: контекст-озод тиллар: 2 тур грамматикаси оркали берилади, замонавий дастурлаш тиллари синтаксис конструкцияси асосида ётади. Тилни англаш вакти кирувчи занжирнинг узунлигидан полиномиал боглик. Квадратик ёки кубик богликлик булиши мумкин. КО-тиллар дастурлаш тилларнинг синтаксис конструкциялари асосида ётади.
Тур 3: регуляр тиллар: энг оддий, англаш вакти кирувчи занжирнинг узунлигидан чизикли боглик. Дастурлаш тилларининг оддий конструкциялари асосида ётади.

4.Контекст-озод грамматикалар.

3-тур грамматикалар лексик таҳлил вақтида ташкил этиладиган белгиларни генерация қилиш учун қулай, лекин улар юқори даража тилларини гапларида бу белгиларни бирлаштириш усулларини ифодалаш учун ноқулай. Масалан, юқорида айтилганидек, 3 тур грамматикаси орқали қавсларни мослаштиришни амалга ошириб бўлмайди. Контекст-озод грамматикалар (2 тур) эса типик дастурлаш тилларининг барча хусусиятларини амалга ошира олмасалар ҳам, универсал ҳисобланадилар ва компиляциянинг синтаксис таҳлил фазаси учун асос сифатида фойдалидирлар.

Контекст-озод грамматикалар компиляциянинг синтаксис таҳлил фазаси учун асос сифатида хизмат қиладилар. Агар қандайдир дастурлаш тилини контекст-озод грамматика ёрдамида генерация қилиш мумкин бўлмаса, у ҳолда шундай контекст-озод грамматикани топиш мумкинки, у ўзига берилган тил дастурини киритувчи тил усти тўпламини генерация қилади.

Синтаксис таҳлил учун бундай грамматикани қўллаш реал тилдаги қатор чегаралар (масалан, дастурдаги барча идентификаторларни эълон қилинишини шартлиги) таҳлилчи орқали текширилмаслигини англатади.

Лекин компиляторда берилган дастурдаги зарур текширувларни бажарилиши билан боғлиқ харакатларни қараб чиқиш қийин эмас (масалан, белгилар жадвалини қўллаш ҳисобига).

Контекст-озод грамматика таърифидан кўринадики, контекст-озод грамматикалар синфи, регуляр грамматикага кўра, жуда қудратли бўлиб (яъни кўп тилларни генерация қилиш мумкин), контекст-боғлиқ грамматикалар синфига нисбатан кучсизроқдир.

$\{a^n b^n \mid n \geq 0\}$ тил контекст-озод ҳисобланади, лекин регуляр эмас. У қуйидаги қоидаларни туғдирувчи грамматика орқали генерацияланади:

$S \rightarrow aSb \mid \varepsilon$

Бошқа томондан $\{a^n b^n c^n \mid n \geq 0\}$ тил контекст-боғлиқ ҳисобланади, контекст-озод эмас.

2. Хомскийнинг нормал формаси.

Контекст-озод грамматикалар бир қатор характеристикаларга эга бўлиб, улар учун қуйидаги холатлар тўғридир:

Каноник форма

а) Ҳар бир контекст-озод грамматика нормал формада Хомский грамматикасига барча қуйидаги кўринишдаги туғилувчи қоидалари билан терминал ва нотерминалларга тегишли оддий ҳолларда эквивалентдир.

$A \rightarrow BC \mid a$

3. **Грейбахнинг нормал формаси.**

б) Ҳар бир контекст-озод грамматика нормал формада Грейбах грамматикасига барча қуйидаги кўринишдаги туғилувчи қоидалари билан эквивалентдир.

$\mathbf{A} \rightarrow ba$

Бу ерда a нотерминаллар қатори (бўш бўлиши ҳам мумкин).

Ўз-ўзига юклаш.

Агар G грамматикада A нотерминал бор бўлса, унинг учун

$A \rightarrow a_1 A a_2$

(бу ерда a_1 ва a_2 терминалларнинг ёки нотерминалларнинг бўш бўлмаган қаторлари ҳисобланиб), у ҳолда бундай грамматика ҳақида ўз-ўзига юкловчи дейилади. Масалан, қуйида келтирилган иккита грамматикалар ўз-ўзига юклашни амалга оширади:

$G_1 = (\{S\}, \{a,b\}, P, S)$ бу ерда P нинг элементлари:

$S \rightarrow aSb$

$S \rightarrow \varepsilon$

$G_2 = (\{S,A\},\{begin,end,[,]\},P,S)$ бу ерда P нинг элементлари:

$S \rightarrow$ begin A end

$S \rightarrow \varepsilon$

$A \rightarrow [S]$

Охирги ҳолда A ва S ҳақида улар ўз-ўзига юклаш хусусиятига эга деб айтилади. Назарий жиҳатдан ихтиёрий ўз-ўзига юклаш хусусиятига эга бўлмаган контекст-озод грамматика регуляр грамматикага эквивалент ва регуляр тилни генерациялайди. Бошқа томондан регуляр грамматика ўз-ўзига юклашга эга бўла олмайди. Ҳақиқатда ўз-ўзига юклаш контекст-озод грамматикани ва регуляр тилларни фарқлашга имкон беради. Иккинчи мисолдан кўриниб турибдики, қавсларни мослашуви ва х.к. лар ўз-ўзига юкловни талаб этади, шунинг учун уларни регуляр грамматика ёрдамида қуриб бўлмайди.

Таҳлил нуқтаи назаридан контекст-озод тил чекли автоматга эквивалент магазин туридаги автоматни англашга ҳақли бўлиб, унга магазин туридаги хотира қўшилганлиги муҳим. Магазин туридаги автоматнинг функцияларига қуйидагилар киради:

а) кирувчи белгини ўқиш, стекнинг юқори белгисини белгилар қаторига ўзгартириш (бўш бўлиши ҳам мумкин) ва ҳолатни ўзгартириш ёки

в) барчаси юқоридагидек, фақат кирувчи белгини ўқимасдан.

Магазин туридаги автоматни қуйидагича қисқача тасаввур қилиш мумкин.

(К, S, Г, δ , S_0 , Z_0)

бу ерда К – холатларнинг чекли тўплами,

S –кирувчи алфавит,

Г- магазин алфавити,

δ -ўтишлар тўплами,

S_0 –бошланғич холат,

Z_0 магазин белгиси,

бошланғич холатда у стекда жойлашади.

Қуйидагича аниқланган М магазин туридаги автоматни мисол сифатида кўриб чиқамиз:

К= {A}, S_0 ={A},

S={‘(‘,’)’}, Z_0 =I

Г= {O,I},

δ δ (A,I,’(‘)=(A,IO) каби берилади.

(бу эса А холатда I билан стек чўққисида ‘(‘ ўқишда А холатга ўтиш кераклигини ва I ни IO га ўзгартиришни билдиради.

δ (A,O,’(‘)=(A,OO), δ (A,O,’)‘)=(A, ε), δ (A,I, ε)=(A, ε)

М автомат мос қавсларни англайди. Очилган қавслар стекка жойланади ва мос ёпилган қавс учрагунча у ерда сақланиб, мос қавс учрагандан сўнг у ердан кетади. Қавслар қатори, агар барча қатор ўқилгандан сўнг стек бўш қолса, қабул қилинади. Бу усулни яна қаторларни охирги холатлар бўйича қабул қиладиган магазин типидаги автоматлар ҳам аниқлашлари мумкин. Ушбу икки тур эквивалентдирлар.

Юқорида ифодаланган магазин туридаги автомат детерминирлангандир, яъни ҳар бир мумкин бўлган кирувчи белгига бир

қийматли ўтиш мавжуд. Чекли автоматлар холига келсак, биз магазин туридаги берилган кириш учун ўтишлар тўпламига, холатига ва стекка эга детерминирланмаган автоматларни худди шундай аниқлашимиз мумкин.

Таҳлил жараёнида мос магазин туридаги автоматни эффектив моделлаштириш амалга оширилади. Баъзи бир контекст-озод тиллар детерминирланган ҳолда таҳлил қилина олмайди (қайтишсиз). Детерминирланган таҳлилга эга тил детерминирланган тил деб аталади, кўпгина дастурлаш тиллари детерминирлангандир, жуда бўлмаса, шунга яқинроқдирлар.

Контекст-озод грамматика ва магазин туридаги автомат ўртасида тўлиқ мослик мавжуд ва автоматнинг детерминирланганлиги тилни генерация қилиш учун қандай грамматикадан фойдаланилаётганлигига боғлиқ. Таҳлил усули детерминирланган (аниқ грамматика учун) ҳисобланади, агар ушбу грамматиканинг таҳлилиида қайтиш талаб этилмаса. Баъзи бир тилларни грамматик таҳлилнинг фақат биргина усули ёрдамида детерминирланган таҳлил этиш мумкин. Хусусий ҳолда, бир қатор тилларни пастдан юқорига, юқоридан пастга эмас, детерминирланган ҳолда таҳлил қилиш мумкин. Бундан кейин биз таҳлилнинг детерминирланган усулларини қараб ўтамиз. Нодетерминирланган усуллар Фортран каби қаторга мўлжалланган тилларга қўлланиши мумкин. Кучли рекурсив тиллар (C++, Паскаль каби) да компиляторнинг жорий қатордагина орқага қайтиши талаб қилинмай, балки дастурнинг каттагина қисмида ҳам талаб қилинса, детерминирланмаган усулларни қўллаш мумкин бўлмайди. Қайтишнинг бошқа камчилиги шундаки, у компиляторнинг синтаксис таҳлил билан параллел олиб бориладиган харакатларини рад этиши мумкин.

Назорат саволлари

1. Хомский иерархияси неча тур грамматикани уз ичига олган? Уларнинг турларини айтинг.
2. Регуляр грамматиканинг ютук ва камчиликлари нималардан иборат?
3. Контекст-озод грамматикага таъриф беринг.

4.Хомскийнинг нормал формаси нима?

5..Грейбахнинг нормал формаси нима?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза № 7.

Мавзу: Разбор муаммоси.

Режа:

1. Разбор муаммоси.

2. Разбор дарахти.

3. Бир хил кийматга эга булмаган грамматикалар.

Калит сузлар.

- **Бошланғич белги**
- **Туғилувчи қоидалар**
- **Чап томонли чиқиш**
- **Ўнг томонли чиқиш**
- **Разбор дарахти**

1. Разбор муаммоси.

Кандайдир тилнинг мавжуд грамматикаси асосида ушбу тилнинг англовчисини куришни англатади.

Разбор масаласи барча тиллар учун ечилмайди.

Разбор масаласи куйидаги масалалар учун ечилади:

- кирувчи белгилар занжирининг маъновий юкланишини аниклаш;
- занжирни куришда фойдаланилган коидаларни аниклаш;
- хатоликларни мавжудлиги фактини аниклаш, хатоликлар урни ва тури.

Компилятор белгилар каторини текшириш муаммосини, ушбу катор шу тилга тегишли ёки йуклигини аниклаш учун ва тегишли булса, у холда тугилувчи грамматика коидалари терминларида каторни структурасини англашни хал килиши керак. Ушбу муаммо разбор муаммоси сифатида машхур. Тугилувчи коидалар билан ишловчи грммматикани текширамиз.

(Е-бошлангич белги).

1. E → E+T
2. E → T
3. T → T*F
4. T → F
5. F → (E)
6. F → x
7. F → y

Куриниб турибдики, (x+y)*x катор ушбу тилга тегишли. Хусусий холда, буни куйидагича келтириб чикариш мумкин (хар бир келтириб чикариш кадами учун кулланилаётган коида раками курсатилган):

2) E → T
3) → T*F
4) → F*F
5) → (E)*F
1) → (E+T)*F
2) → (T+T)*F
4) → (F+T)*F
6) → (x+T)*F
4) → (x+F)*F
7) → (x+y)*F
6) → (x+y)*x

Ёки куйидагича келтириб чикариш мумкин:

2) E → T
3) → T*F
6) → T*x
4) → F*x
5) → (E)*x
1) → (E+T)*x

4) → (E+F)*x
7) → (E+y)*x
2) → (T+y)*x
4) → (F+y)*x
6) → (x+y)*x

Биринчи чикишнинг хар бир боскичида сентенциал форманинг энг чап нотерминали грамматиканинг бирон бир тугилувчи коидаси ёрдамида алмаштирилди. Шу сабабли ушбу чикиш **чап томонли чикиш** дейилади. Хар бир боскичида энг унг нотерминал алмаштирилган иккинчи чикиш эса **унг томонли чикиш** деб аталади.

Шу каби бошка чикишлар хам мавжудки, улар чап томонли хам, унг томонли хам хисобланмайдилар, лекин трансляторларни куришда улардан фойдаланилмайди. **Гапни чап томонли разбори** чап томонли чикишни кулланилган холда гапни генерация килиш учун тугилувчи коидаларнинг кетма-кетлиги сифатида аникланади. Ушбу холда чап томонли разборни куйидагича ёзиш мумкин:

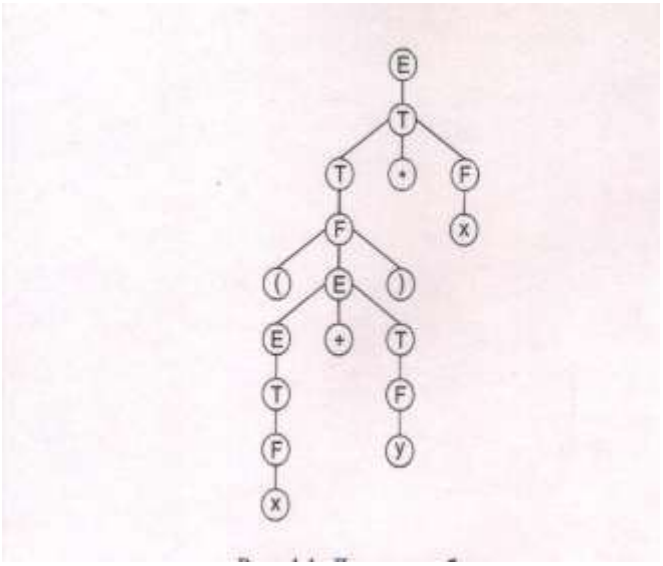
2,3,4,5,1,2,4,6,4,7,6.

Гапни унг томонли разбори унг томонли чикишни кулланилган холда гапни генерация килиш учун тугилувчи коидаларнинг тугилувчи коидаларнинг тескари кетма кетлиги хисобланади; масалан, юкорида келтирилган холда унг томонли разбор куйидагича ёзилади:

6,4,2,7,4,1,5,4,6,3,2.

Тугилувчи коидалар кетма-кетлигининг тескари тартиби шу билан богликки, унг томонли разбор гапни бошлангич белгига келтириш сифатида каралади, гапни бошлангич белгидан бошлаб генерация килиш эмас (пастдан юкорига караб разбор). Шуни таъкидлаш керакки, хар бир тугилувчи коидадан иккала чикишда хам бир хил сон марта фойдаланилади (разборларда).

2.Разбор дарахти. Чикиш яна синтаксис дарахт (разбор дарахти) номи билан машхур дарахтни куриш терминларида хам ифодаланиши мумкин. (x+y)*x катор холида синтаксис дарахт куйидаги куринишда булади:



Расм 1. Разбор дарахти.

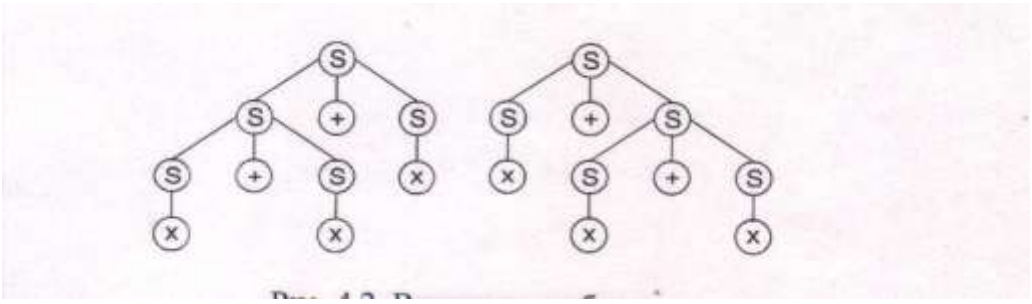
Разбор муаммисини куйидаги масалаларга келтириш мумкин.

1. чап томонли разборни топиш
2. унг томонли разборни топиш
3. синтаксис дарахтни куриш.

3.Бир хил кийматга эга булмаган грамматикалар.

Купгина холларда чап томонли ва унг томонли разборлар ва синтаксис дарахт уникал хисобланади. Лекин, куйидаги тугилувчи коидали грамматика учун

S → S+S| x x+x+x гап иккита синтаксис дарахтга, иккита чап (унг) томонли разборга эга.



Расм 2. Разбор вариантлари.

S → S+S
→ S +S+S
→ x+S+S
→ x+x+S
→ x+x+x

S → S+S
→ x+S
→ x+S+S
→ x+x+S
→ x+x+x

Агар грамматикада генерация килинган кандайдир гап, биттадан ортик разбор дарахтига эга булса, бундай грамматика хакида у бир кийматли эмас дейилади. Эквивалент шарт шунда куринадики, гап биттадан ортик чап ёки унг томонли разборга эга булиши керак. Грамматиканинг бир кийматли эмаслигини урнатиш масаласи умумий холда ечими йук масаладир, яъни киришда ихтиёрий грамматикани кабул киладиган ва уни бир кийматлими ёки йуклигини аниклайдиган универсал алгоритм мавжуд эмас. Баъзи бир бир кийматли булмаган грамматикаларни уша тилни генерация киладиган бир кийматлига айлантириш мумкин. Масалан, куйидаги тугилувчи коидаларга эга грамматика

S x S +x бир кийматли булиб, у уша тилни худди аввалги бир кийматли булмаган грамматика каби генерациялайди.

Разбор усуллари купинча пастдан юрувчидир, яъни бошлангич белгидан бошлаб гапга караб юрилади ёки юкоридан юрувчи булиб, гапдан бошлаб бошлангич белгига караб юрилади.

Разборда ечимни рад этиш (отказ) кайтиш (возврат) деб аталади. Разбор усуллари кайтиш бор ёки йуклигига караб детерминирован ва нодетерминирован булиши мумкин. Нодетерминирован усуллар хотира ва вакт нуктаи назаридан киммат булиб, компиляция вактида бажарилувчи натижалари кейинчалик йук килиниши керак булган харакатларни синтаксис анализаторга кушишни кийинлаштиради (масалан, белгилар жадвалини куриш ва х.к.). Бундан сунг биз факат разборнинг детерминирован усуллари хакида суз юритамиз.

Назорат саволлари

1. Разбор муаммоси нима?
2. Чап томонли ва унг томонли разбор нима?
3. Нима учун асосий эътибор, жуда куп сонли чап ёки унг томонли булмаган разборлар була туриб, чап ва унг томонли разборларга каратилган?
4. Синтаксис дарахтни куришда разбор муаммосини кандай хал этиш мумкин?
5. Бир кийматли булмаган грамматикага таъриф беринг.
6. Детерминирован ва нодетерминирован разбор усулларини фаркини айтиб беринг

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.

2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с

3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.

4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.

5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.

6. WWW.codecrojekt.ru

7. WWW. master.ru

8. WWW.bdn_borland.com

9. http://microsofft.com

Маъруза 8.

Мавзу: Лексик тахлилчилар.

Режа:

1. **Лексик тахлил масаласи ва унинг ечиш йуналишлари;**
2. **Лексик тахлилнинг хизматчи жадваллари;**
3. **Идентификаторлар жадвали;**
4. **КЭШ манзиллаш**

Калит сузлар.

- **Сканер**
- **Лексема**
- **Идентификаторлар жадвали**
- **Терминал белгилар жадвали**
- **Константалар жадвали**
- **Лексема кодлари жадвали**

1.Лексик тахлил масалалари ва уларнинг ечиш йуналишлари;

Лексик тахлил уз ичига куйидаги масалаларни ечишни олади:

1. дастурнинг берилиш матнини караб чикиш (сканирование);
2. изохлар ва кераксиз пробелларни йукотиш;
3. лексик улчовларни ажратиш (лексик улчовларни чегараларини топиш);
4. лексемларни ички тасвирлаш куринишларига айлатириш;
5. лексик улчовларни англаш;
6. лексик анализаторнинг чикувчи жадвалларини ташкил этиш.

Купгина дастурлаш тилларида лексемларни чегаралари булиб куйидагилар хисобланадилар:

- пробеллар;
- булакловчилар (разделители)(. , : ;);
- амал ишоралари (+, -, :, *, :=);

Оддий холда лексема – бу чегаралар орасида жойлашганлардир.

Константа ва исмлар каби лексемаларни англаш учун 3 синф регуляр грамматикасидан фойдаланилади ва англовчи тугалланган автоматни ташкил этади.

Скаринлаш натижасида дастур узун катордан ташкил топади.

Лексик тахлилчини амалга ошириш учун икки йуналиш мавжуд.

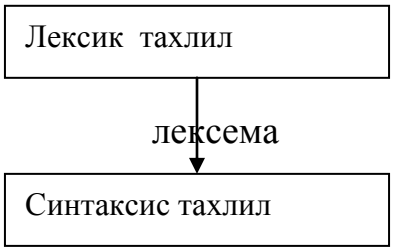
Биринчи томон – лексик ва синтаксис анализаторларнинг узаро харакатларини амалга оширишнинг икки йуналиши.

Иккинчи – лексик анализаторнинг курилиши.

Биринчи йуналиш : ва синтаксис анализаторларнинг узаро харакатларини амалга оширишнинг охирги схемаси. Лексик анализатор барча дастурни караб чикади ва барча лексемлар учун кодлар жадвалини тузади ва натижаларни синтаксис анализаторга узатади. Бу холатда лексик ва

синтаксис анализаторлар булакланган хисобланадилар ва компилятор фазасида бир биридан кейин бажариладилар.

Параллел йуналишда лексик анализатор навбатдаги лексемани топади ва уни синтаксис анализаторга узатади.



Бу холатда лексик анализатор навбатдаги лексемани ажратиш учун синтаксис анализатордан чакириладиган кисм дастурдан ташкил топади.

Кетма-кет схеманинг устунлиги унинг тезрок ишлашидадир.

Кетма-кет схеманинг камчилиги хотиранинг куп сарфланишидир.

Параллел схеманинг устунлиги: амалга оширишнинг оддийлиги, кам хотира сарфланишидир.

Параллел схеманинг камчилиги: компиляция вакти 1,5 – 2 маротаба ошади.

Иккинчи томон.

Биринчи йуналиш: лексемалар сифатида алфавитнинг алохида белгилари олинади: харфлар, ракамлар, булакловчилар, амалларнинг ишоралари ва х.к. Бу холда калит суз, масалан, BEGIN бешта лексеманинг туплами сифатида каралади: ‘B’, ‘E’, ‘G’, ‘I’, ‘N’. Бундай йуналишда купсимволлик лексемаларни кайта ишлашда (синтаксис тахлил боскичида) компиляция вакти синтаксис тахлилнинг мураккаблиги хисобига купайиб кетади. Иккинчи йуналиш: алохида калит сузлар, амалларнинг ишоралари, исмлар, константалар алохида лексема куринишида каралади ва лексик тахлил масаласи уларни англашдан иборат булади. Бундай йуналишдан куп фойдаланилади.

2.Лексик тахлилнинг хизматчи жадваллари.

Лексик тахлил учун бошлангич маълумотлар булиб белгилар катори куринишида ифодаланган дастур хисобланади.

Жадваллар куйидаги гурухларга булинади:

1. кирувчи;
2. оралик;
3. чикувчи;

Кирувчи – терминал белгилар жадвали. У ёзувлардан ташкил топади:

1	2	3	4
Ракам	Белги	Тури	Приоритет белгиси

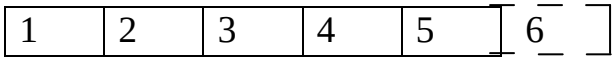
Амал килиш нуктаи назаридан - бу ёзувлар туплами ёки руйхат.

Майдонлар таркиби буйича:

1. Тартиб раками;

2. Белгининг узи;
3. Терминал белгининг тури (калит суз, ажратувчи, амал ишораси);
4. Арифметик ва мантикий амаллар учун приоритет белгиси. (лексик анализатордан кейин бошка боскичларда тулдирилади);

Константалар (литераллар) жадвали куйидаги ёзувлар тупламини ифодалайди:



1 – тартиб раками;
2 – константалар киймати;
3 – асос (сонли константалар учун);
4 – анкилик (хотира улчами);
5 – константа тури;
6 – кушимча маълумот, масалан, литерал учун – хотирадаги жойлашувнинг бошлангич манзили.

3.Идентификаторлар жадвали

Идентификаторлар жадвали – кирувчи дастур элементлари хакидаги маълумотларни сакловчи берилганлар туплами. Бир неча хил идентификаторлар жадвали мавжуд.

ИЖ ташкил этиш: Идентификаторга боглик холда тупламда турлича маълумотлар сакланади. 1)Узгарувчи-исм,тур,адрес 2)Константа-исм, агар бор булса,адрес,киймат;3)Функция аргументлар сони,тур,исм,адрес

ИЖ лексик тахлил фазасида ташкил этилади ва кетма-кет тулдирилади. Лексик тахлил фазасида идентификаторлар турлари ёзилади. Кодни генерациялашга тайёргарлик фазасида – хотира ажратилади.

ИЖ ини ташкил этишнинг асосий критерийси булиб кидирув вакти хисобланади, чунки асосий функциялар ёзувлврни кидирув ва кушишдан иборат (купрок кидирув амалга оширилади)

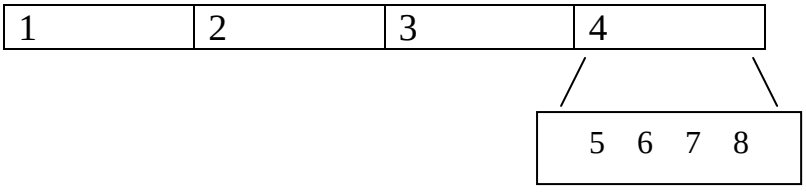
I. Тартибланмаган руйхат – ИЖ да элементлар келиб тушиши буйича кушидали.

II. Тартибли руйхат – Иждаги барча элементлар табиий тартибга мос равишда ошиб бориш ва камайиш буйича тартибланган.

III. ИЖ ининг бинар дарахти бинар дарахт куринишига эгадир. Хар бир элемент ёки чукки -идентификатордир. Дарахтнинг илдиз чуккиси булиб, биринчи келиб тушган идентификатор хисобланади.

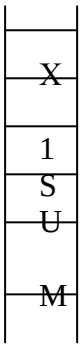
Бинар дарахтнинг формаси келиб тушаётган идентификаторларга боглик. Камчилиги дарахтдан тартибли руйхатни келитириб чикариш мумкин.

Идентификаторлар жадвалига барча калит суз хисобланмаган исмлар киритилади. У куйидаги куринишга эга:



3,4,5,6,7,8 – бошка боскичларда тулдирилади;
1 – тартиб раками;
2 – исм узунлиги байтларда;
3 – хотирадиги исм белгиларини саклаш манзили;
4 – идентификатор атрибутлари;
Атрибутлар сифатида:
5 – куриниш (ёки оддий узгарувчи исми, ёки туплам, ёки процедура (функция), метка);
6 – тур (хакикий, бутун, каторли, белгили, мантикий);
7 – хотира синфи (статик, автоматик, динамик, ташки);
8 – улчови (масалан, катор учун – узунлик, туплам учун – индекслар сони, интервьюлашган турлар учун – чап ва унг чегаралар киймати);
Маълумотларни исмларда тасвирлашнинг шундай вариант мавжудки унда барча исмни ташкил этувчи символлар занжирни ташкил этадилар ва хар бир янги исм Ушбу занжирнинг охирига уланадилар.
Исмлар жадвали

Ракам	Исм узунлиги	Бошлангич манзил
1	2	
2	3	



Var
X1, sum: integer

Чикувчи жадвал – лексемалар кодлари жадвали (синтаксис тахлил учун бошлангич).
Жадвал структураси:

Лексема раками (дастурда)	Тури	Лексема раками (жадвалда)
1		
2		

Биринчи майдонда – дастурдаги лексемани пайдо булиш раками.
Учинчи майдонда – мос жадвалдаги ракам. Агар исм бир неча марта кайтарилса, у холда лексемалар кодлари жадвалида унга шунча марта катор мос келади.
Иккинчи майдонга исм хакидаги маълумот киритилади: исм – I, терминал белгилар – T, константалар – C.
Жадвални ташкил этишга мисол караймиз.

```
Program authm;
Var
        I, J, Sum: Integer;
Begin
        Sum:=0;
        For i:=1 to 100 do;
```

```
    Begin
        Read(J);
        Sum:=Sum+J;
    End
    Sum:=Sum div 100;
    Write(Sum);
End
```

Терминал белгилар жадвали

№	Белги	Тури		
		Ажратувчи	Амал ишораси	Калит Суз
1	**;**	1	0	0
2	**,**	1	0	0
3	**:**	1	0	0
4	пробел	1	0	0
5	**:=**	0	1	0
6	**(**	1	0	0
7	**)**	1	0	0
8	+	0	1	0
9	Div	0	1	0
10	Program	0	0	1
11	Var	0	0	1
12	Integer	0	0	1
13	Begin	0	0	1
14	For	0	0	1
15	To	0	0	1
16	Do	0	0	1
17	End	0	0	1
18	Read	0	0	1
19	Write	0	0	1
20	**.**	1	0	0
21	-	0	1	0
22	*	0	1	0

Исмлар жадвали

№	Исм
1	Arithm
2	I
3	J
4	Sum

Константалар жадвали

№	Константа	Асос0	Тури	Аниклик
1	0	10	Бутун	2
2	1	10	Бутун	2
3	100	10	Бутун	2

Лексемалар коди

№	Тури	№ жадвалдаги раками	Лексемалар
1	T	10	Program
2	T	4	Пробел
3	I	1	Arithm
4	T	1	;
5	T	11	Var
6	T	4	Пробел
7	I	2	I
8	T	2	,
9	I	3	J
10	T	1	;
11	…	…	…

Лексема кодларининг чикувчи жадвалида пробел хакидаги маълумот жойлашмайди.

4.КЭШ манзиллаш

Усулга кура белги тупламда элементни индексига айлантирилади. КЭШ лаш – белги устида ихтиёрий мантикий ёки арифметик амални бажариш демакдир. КЭШ манзиллаш КЭШ функция томонидан ячейка адреси сифатида кандайдир берилганлар тупламидан кайтарилаётган кийматдан фойдаланишдир. Берилганлар туплами улчови КЭШ функйиянинг кийматлари сохасига мос келиши керак.

Кидирув вакти ва жойлаштириш вакти – бу КЭШ функцияни хисоблаш учун сарф булган вактдир.

Камчиликлари: 1)Хотира хажмидан ноэффектив фойдаланиш; 2)КЭШ функцияни мос тугри танловини зарурлиги.

Коллизия – бу иккита турли идентификаторга битта КЭШ функциянинг киймати мос келган холатдир.

Коллизияни ечиш усуллари (рекэшлаш). Коллизия юз берганда кэш функциянинг янги киймати хисобланади.

Рекэшлаш формуласи $h_i(A)=(H(A)+p_i)\text{mod}N_m$ N_m – кэш лаш сохасининг максимал киймати, p_i хисобланган сон

Оддий рекэшлаш усули $h_i(A)=(h(A)+i)\text{mod}N_m$ (p_i урнига i ни оламиз)

Псевдотасодифий сонлардан фойдаланган холда рекэшлаш p_i урнига бутун псевдотасодифий сонларнинг кетма-кетлигини оламиз. Псевдотасодифий сонлар генераторини муваффакиятли танлашда $k=N_m$

Купайтириш ёрдамида рекэшлаш $h_i(A)=(h(A)*i)\text{mod}N_m$

Занжирлар усули Бу ерда ИЖ га кэш жадвал кушилади, унда ёки буш киймат, ёки асосий ИЖ идаги кандайдир хотира сохасига курсаткични киймати сакланади.

Ютуклари: 1)ИЖ ини буш кийматлар билан тулдириш зарурияти йук.2) хар бир идентификаторга ИЖ ида катъий битта ячейка мос келади, чунки унда

буш ячейкалар йук. ИЖ да ссылкалар майдони кушилади, унда ИЖ нинг ихтиёрий элементига курсаткич мавжуд булиши мумкин. Асосий ИЖ ининг биринчи озод ячейкасига курсатувчи махсус узгарувчидан фойдаланилади.

Комбинирланган усуллар. Агар коллизиялар юз бермаса ссылкани кушимча майдонидан фойдаланилади, маълумотларни танлаш учун кэш функциядан фойдаланилади ва ссылки майдони буш колади. Агар коллизия юз берса, у холда ссылки майдони оркали идентификаторларни кидируви юкорида курилган усуллардан бир буйича ташкил этилади.

Занжирлар устидаги ютуклар: *кэш функциянинг мос тушувчи кийматли идентификаторларини саклаш учун асосий ИЖ билан кесишмайдиган хотира сохасидан фойдаланилади.*

Камчиликлар: хотиранингдинамик таксимланган сохалари билан ишлаш зарурияти.

Назорат саволлари.

1. Лексема нима?
2. Лексик тахлил килиш нима учун керак?
3. Лексик тахлилнинг хизматчи жадваллари нима учун керак?
4. Идентификаторлар жадвали нима?
5. Исмлар жадвали нима?
6. Терминаллар жадвали нима?
7. КЭШ манзиллаш нима?
8. Константалар жадвали нима?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. Компаниец Р.И. Маньков Е.В. Филиппов Н.Е. Системное программирование, основы построения трансляторов. Учебное пособие для средних и высших учебных заведений. СПб: Карона, 2000-256с.
7. WWW.codecrojekt.ru
8. WWW. master.ru
9. WWW.bdn_borland.com
10. http://microsofft.com

Маъруза № 9.
Мавзу:Чекли автоматлар.
Режа:
1.Чекли автоматлар.
2.Чекли детерминирланган автоматлар.
3.Чекли нодетерминирланган автоматлар.

Калит сузлар.
- **Чекли автомат**
- **Холатларнинг чекли тўплами**
- **Чекли кириш алфавити**
- **Ўтишлар тўплами**
- **Бошланғич холат**
- **Охирги холатлар тўплами**
- **Детерминирланган автомат**
- **Нодетерминирланган автомат**

1.Чекли автоматлар.

Қуйидагича аниқланадиган регуляр ифодалар, грамматика турлари ва чекли автоматлар ўртасида тулик мослик мавжуд:

Чекли автомат - бу қандайдир тилни қаторларини англаш учун қурилмадир. Унинг таркибида чекли ҳолатлар тўплами бор бўлиб, уларнинг баъзилари охиргилари деб аталади. Ҳар бир литеранинг ўқилиши давомида назорат қатори ҳолатдан ҳолатга берилган ўтишлар тўпламига мос равишда узатилади. Агар қаторнинг охирги литерасини ўқиганидан сўнг автомат охирги ҳолатлардан бирида бўлса, қатор ҳақида, у автомат билан қабул қилинган тилга тегишли деб айтилади. Бошқа ҳолларда қатор автомат билан қабул қилинган тилга тегишли эмасдир.

Чекли автомат формал равишда қуйидаги бешта характеристика билан аниқланади:

- (К) холатларнинг чекли тўплами
- (Σ) чекли кириш алфавити
- (δ) ўтишлар тўплами
- ($S_0 \in К$) бошланғич ҳолат
- ($f \in К$) охирги ҳолатлар тўплами

$M = (K, \Sigma , \delta , S_0 , f)$.

2.Чекли детерминирланган автоматлар.

Мисол: Автоматнинг ҳолатлари A ва B, кирувчи алфавит {0,1}, бошланғич холати –A , охирги ҳолатлар тўплами -{A}, ўтишлар

(A,0) = A, (B,0)=B

(A,1) = B, (B,1)=A

Бу ўтишлар A ҳолатда 0 ни ўқишда бошқарув A ҳолатга узатилишини англатади ва х.к. 01001011 қаторни ўқишда бошқарув кетма-кет равишда қуйидаги тартибда узатилади: A,A,B,B,B,A,A,B,A

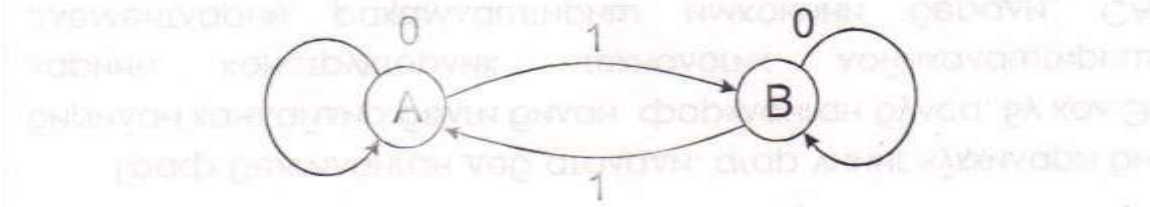
A сўнгги ҳолат бўлгани учун қатор чекли автомат томонидан қабул қилинади, лекин 00111 қаторни ўқишда автомат қуйидаги ҳолатлар орқали ўтади A,A,A,B,A,B

B охирги ҳолат бўлмаганлиги сабабли, бу қатор қабул қилинмайди, яъни бу қатор ушбу автомат қабул қиладиган тилга тегишли эмас. Шу сабабли, нуллар автоматнинг ҳолатига таъсир кўрсатмайди, ҳар бир 1 эса унинг ҳолатини A дан B га B дан A га ва бошланғич ҳолат эса охирги ҳолатдек бўлади, яъни автомат билан қабул қилинган тил жуфт бирлар сонига эга қаторлардан тузилган.

Ўтишларни қуйидаги жадвал (9-жадвал) орқали ва чизма кўринишини (16- расм) тасаввур қилиш мумкин

9- жадвал.

Ҳолатлар			
Кириш		A	B
	0	A	B
	1	B	A



16- расм. Чекли детерминирланган автоматнинг ўтишлари

Бундай автоматни детерминирланган деб аталади, чунки ўтишлар жадвалининг ҳар бир элементида битта ҳолат мавжуд. Детерминирланмаган чекли автомат ҳолида эса ушбу шартга амал қилинмайди.

3.Чекли нодетерминирланган автоматлар.

Қуйидаги ҳолат орқали аниқланган чекли автоматни қараймиз:

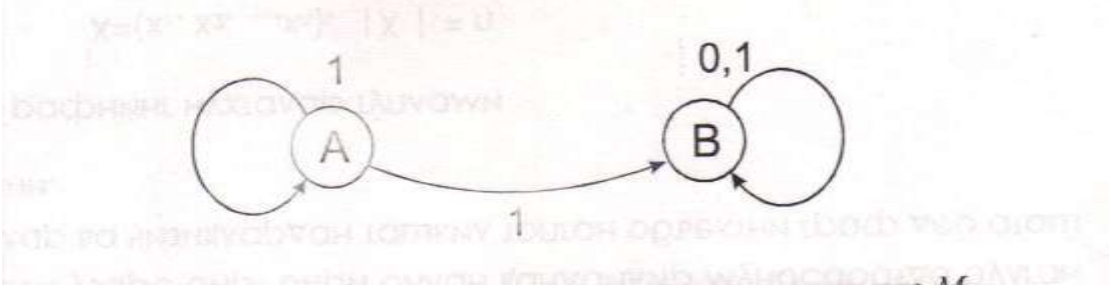
$M_1 = (K_1, \Sigma_1, \delta_1 , S_1 , f_1)$.

Бу ерда $K_1=\{A,B\}$, $\Sigma_1=\{1,0\}$, $S_1=\{A\}$, $f_1=\{B\}$

Ўтишлар эса 10- жадвал ва 17- расмда келтирилган.

10-жадвал

Ҳолатлар			
Кириш		A	B
	0	θ	{B}
	1	{A,B}	{B}



17- расм. M1 чекли детерминирланмаган автоматнинг ўтишлари.

Биринчи қатор қабул қилинади, чунки қаторни ўқишда охирги ҳолатга элтувчи ўтиш мавжуд (ўтишлар кетма-кетлиги). Худди шундай охирги бўлмаган қаторга ҳам ўтиш мавжуд, лекин бунинг қаторнинг қабул қилинишига аҳамияти йўқ. Шу сабабли, қатор детерминирланмаган чекли автомат томонидан қабул қилинмаслиги мумкин деган хулосага келишдан аввал, барча мавжуд ўтиш имкониятларини кўриб чиқиш зарур.

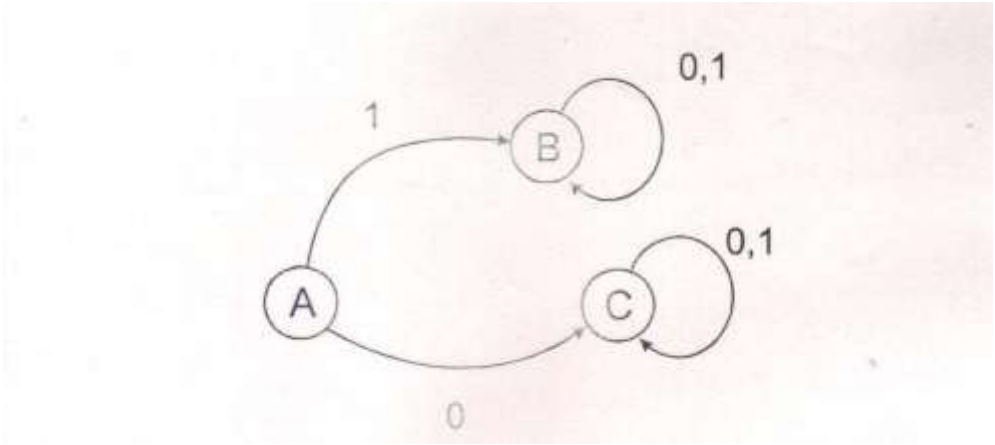
M2 чекли детерминирланган M1 га мос келувчи ўша тилни қабул қилувчи автомат мавжуд. M2 автоматнинг ўтишлари 11- жадвал ва 18-расмда келтирилган.

$M_2 = (K_2, \Sigma_2, \delta_2 , S_2 , f_2)$.

Бу ерда $K_2=\{A,B,C\}$, $\Sigma_2=\{1,0\}$, $S_2=\{A\}$, $f_2=\{B\}$

11-жадвал.

Ҳолатлар				
Кириш		A	B	C
	0	C	B	C
	1	B	B	C



18-расм. M2 детерминирланган чекли автоматнинг ўтишлари.

М1 каби М2 қатор хам 0 ва 1 лардан ташкил топган қаторларни қатор 1 дан бошланган ҳолда қабул қилади. Лекин қаторни М2 ёрдамида англашда қайтиш ҳеч қачон талаб этилмайди, чунки аниқ бир кирувчи белгини ўқиш жараёнида ихтиёрий ҳолатдан бошқа ҳолатга аниқ бир ўтиш амалга оширилади. Бу эса М2 дан фойдаланишда қаторни англаш вақти унинг узунлигига пропорционал бўлади.

Лексик таҳлилга мослик шундан иборатки, 3 турдаги ҳар бир тилга детерминирланган чекли автомат мос келиб у ушбу тилни қаторларини англаш вазифасини бажаради. Масалан, G1 грамматикада генерация қилинган ва қуйидаги туғилувчи қоидалар ёрдамида аниқланган қаторлар

$$A \rightarrow 1A|\ 1B\ |\ 1$$

$$B \rightarrow\ 0B|\ 1B\ |\ 0\ |\ 1$$

Бу ерда A бошланғич белги бўлиб, М1 ва М2 ёрдамида англанади.

Грамматикани детерминирланмаган чекли М1 автоматдан қуйидагича олинади:

1. Автоматнинг бошланғич ҳолати грамматика гапининг бошланғич белгиси бўлади.
2. $\delta(A,1) =A,$ $\delta(B,0)=B$

   $\delta(A,1) =B,$ $\delta(B,1)=B$

ўтишларга қуйидаги туғилувчи қоидалар мос келади:

$A \rightarrow 1A|\ 1B$

$B \rightarrow\ 0B|\ 1B$

А ҳолатда 1 ни ўқишда охирги В ҳолатга ўтиш $A \rightarrow\ 1$ га мос келади ва аналогик равишда $B \rightarrow\ 0 | 1$

Худди шундай, грамматикадан М1 автоматни келтириб чиқариш мумкин.

Назорат саволлари

1. Чекли автоматлар Хомский иерархияси буйича кайси грамматика турига тегишлидир?
2. Чекли автоматга тариф беринг.
3. Детерминирланган чекли автомат нодетерминирланган чекли автоматдан кандай фарк килади?
4. Регуляр ифодани бир кийматли чекли детерминирланган автоматга айлантириш мумкинми?

Фойдаланилган адабиётлар

1. Карпов С.Ю. Теория автоматов. Учебные пособия для вузов. –СПб: Питер, 2003.-201с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза № 10.
Мавзу: Синтаксис тахлилчилар.
Режа: 1Синтаксис анализаторларнинг асосий ишлаш принциплари.
2. Амаллар дарахти.
3.Синтаксис тахлилчиларни қуришни автоматлаштириш.

Калит сўзлар.

- **Амаллар дарахти**
- **Жадвалли-бошқарувчи дастур**
- **Дастурли –бошқарилувчи дастур**
- **Мақсадга йўналтирилган грамматик тахлил**

1.Синтаксис тахлилчиларнинг асосий ишлаш принциплари.

Асосий функциялар:

1) асосий синтаксис конструкцияларни топиш ва ажратиш.

2) ҳар бир конструкцияни турини ўрнатиш ва текшириш.

3) матнни генерация қилиниши учун қулай синтаксис конструкция кўринишида тасвирлаш.

2. Амаллар дарахти.

Синтаксис дарахтда барча чўққилар амалларга мос келади, барглар операндларга мос келади ва синтаксис дарахт барглари идентификаторлар жадвалининг ёзувлари билан боғликдир.

3. Синтаксис тахлилчиларни қуришни автоматлаштириш.

Синтаксис тахлилчини куриш учун иккита турли усуллардан фойдаланиш мумкин. Улардан бири – бу грамматик разборнинг барча мумкин булгшан грамматикалар учун керак буладиган универсал дастурини ёзишдир. Бу холда аник грамматикалар ушбу дастурга уни ишини бошкурувчи кандайдир структура куринишида берилади. Шунинг учун бундай дастур **жадвалли-бошкарувчи дастур** деб аталади.

Бошка усул – аник берилган тил учун махсус грамматик разбор учун ишлаб чикилган дастурдир; бу холда уни синтаксиси аник коидалар буйича операторлар кетма-кетлигида ифодаланади, яъни дастурда. Разборнинг

бундай амалга оширилишини **дастурли –бошкарилувчи дастур** деб атаймиз.

Ушбу усулларнинг хар бири узининг ютук ва камчиликларига эга. Аник бир тил учун трансляторни куришда универсал дастурга хос булган юкори эгилувчанлик ва параметрлаш талаб этилмайди. Берилган тил учун махсус ёзилган грамматик разборнинг дастури купинча фойдалирок булади ва у билан ишлаш осонрок кечади. Ихтиёрий холда хам берилган синтаксисни **синтаксис граф куринишида** ифодалаш фойдалидир. Бундай граф гапларни грамматик тахлил ишида бошкарув жараёнини ифодалайди.

Пастдан чикувчи грамматик разбор учун тахлил максади аввалдан маълум булади. Бу максад- гапни англаш, яъни бошлангич белгидан тугилувчи белгилар кетма-кетлигини англашдир. Бу усулни яна **максадга йуналтирилган грамматик разбор** деб хам аталади. Грамматик разборнинг дастурини куришда нотерминал белгилар ва максадларнинг мослигидан фойдаланиш максадга мувофикдир, хар бир нотерминал белги учун узининг грамматик разбори процедураси курилади. Хар бир бундай процедуранинг максади гапнинг мос нотерминал белгидан тугилиши мумкин булган булагини англаш.

Назорат саволлари.

1. Жадвалли –бошкарувчи ва дастурли бошкарувчи синтаксис тахлилчиларнинг ютук ва камчиликларини айтинг.
2. Тахлилнинг кандай усулини максадга йуналтирилган деб хам аташ мумкин?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза №11.

Мавзу: Синтаксис графни куриш.

Режа: 1.Синтаксис графни куриш.

2.Синтаксис графни куриш коидалари

3.Берилган синтаксис учун грамматик таҳлил дастурини қуриш.

4. Грамматик тахлилнинг жадвалли-бошқарув дастурини қуриш.

Калит сузлар.

- **Синтаксис граф**
- **Қобиқ**
- **Туғилувчи қоидалар**

1.Синтаксис графни куриш.

Синтаксис таҳлилчини қуриш учун иккита турли усуллардан фойдаланиш мумкин. Улардан бири – бу грамматик таҳлилнинг барча мумкин бўлган грамматикалар учун керак бўладиган универсал дастурини ёзишдир. Бу ҳолда аниқ грамматикалар ушбу дастурга уни ишини бошқарувчи қандайдир структура кўринишида берилади. Шунинг учун бундай дастур **жадвалли-бошқарувчи дастур** деб аталади.

Бошка усул – аниқ берилган тил учун махсус грамматик таҳлил учун ишлаб чиқилган дастурдир; бу ҳолда уни синтаксиси аниқ қоидалар бўйича операторлар кетма-кетлигида ифодаланади, яъни дастурда. Таҳлилнинг бундай амалга оширилишини **дастурли –бошқарилувчи дастур** деб атаймиз.

Ушбу усулларнинг ҳар бири ўзининг ютуқ ва камчиликларига эга. Аниқ бир тил учун трансляторни қуришда универсал дастурга хос бўлган юқори эгилувчанлик ва параметрлаш талаб этилмайди. Берилган тил учун махсус ёзилган грамматик тахлилнинг дастури кўпинча фойдалирок бўлади ва у билан ишлаш осонроқ кечади. Ихтиёрий ҳолда ҳам берилган синтаксисни **синтаксис граф кўринишида** ифодалаш фойдалидир. Бундай граф гапларни грамматик таҳлил ишида бошқарув жараёнини ифодалайди.

Пастдан чиқувчи грамматик таҳлил учун таҳлил мақсади аввалдан маълум бўлади. Бу мақсад- гапни англаш, яъни бошланғич белгидан туғилувчи белгилар кетма-кетлигини англашдир. Бу усулни яна **мақсадга йўналтирилган грамматик таҳлил** деб ҳам аталади. Грамматик таҳлилнинг дастурини қуришда нотерминал белгилар ва мақсадларнинг мослигидан фойдаланиш мақсадга мувофиқдир, ҳар бир нотерминал белги учун ўзининг грамматик таҳлили процедураси қурилади. Ҳар бир бундай процедуранинг мақсади гапнинг мос нотерминал белгидан туғилиши мумкин бўлган бўлагини англаш. Биз грамматик таҳлилнинг барча дастурини графини

қуриш мақсадида эканмиз, у ҳолда ҳар бир нотерминал белги қисм граф кўринишида ифодаланади.

Синтаксис графни қуриш қоидалари:

А1. Ҳар бир нотерминал белги А мос туғилувчи қоидалари тўплами билан

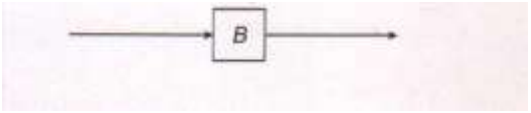
$A::=\beta_1|\ \beta_2|\ \ldots.|\ \beta_n$

А синтаксис графда ўнг қисми А2-А6 қоидаларга мос равишда аниқланган структура асосида ифодаланади.

А2. β_i даги ҳар бир x терминал белгининг пайдо бўлиши ушбу белгини кирувчи гапда англаш операторига мос келади. Графда бу айлана ичига олинган x белги кўринишидаги қобиқ (1-расм) сифатида акс этади.

1-расм.

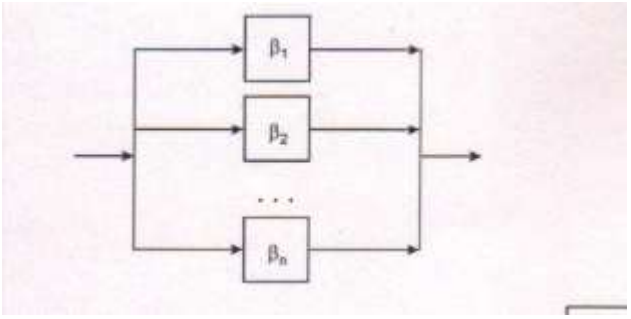
А3. β_i даги ҳар бир В нотерминал белгининг пайдо бўлиши В англаш процедурасига мурожатга мос келади. Графда бу тўғритўртбурчак ичига олинган В белги кўринишидаги қобиқ (2 –расм) сифатида акс этади.



2-расм.

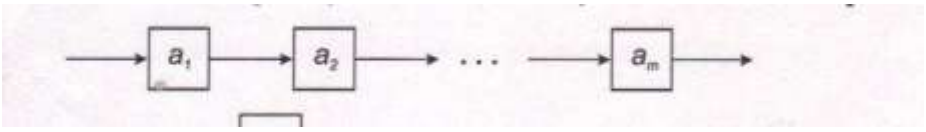
А4: Қуйидаги кўринишдаги туғилувчи қоидалар $A::=\beta_1|\ \beta_2|\ \ldots.|\ \beta_n$

графда (3-расм) қуйидагича акс этирилади:



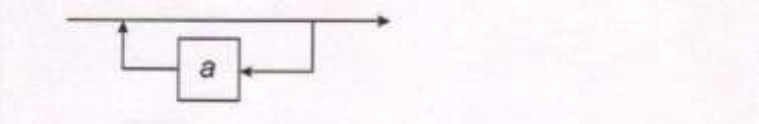
3-расм.

А5: $\beta = \alpha_1, \alpha_2, \ldots, \alpha_m$ қуйидаги кўринишга эга β қатор графда (4-расм) қуйидагича ифодаланади.



4-расм.

А6: $\beta = \{\alpha\}^*$ қуйидаги кўринишга эга β қатор графда (5–расм) қуйидагича ифодаланади.



5-расм.

Мисол.

A::=x| (B),

B::=AC

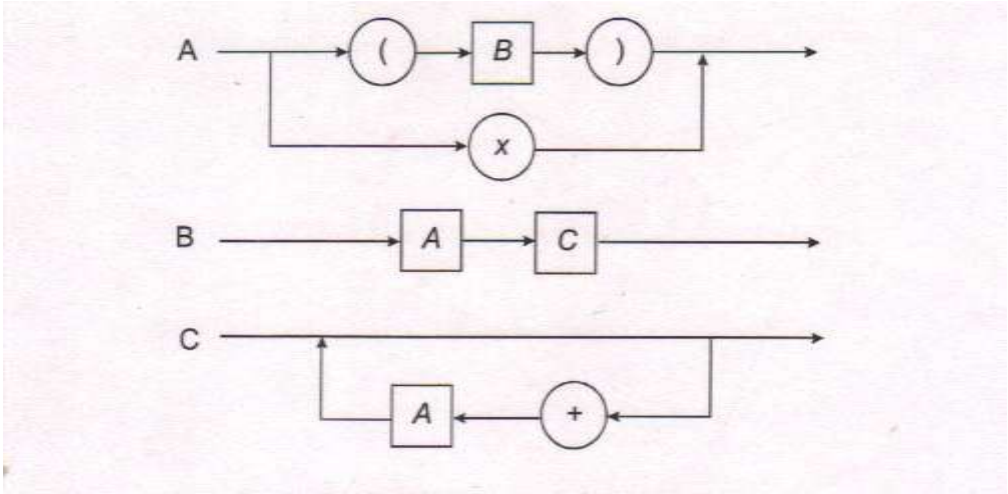
C::={+A}*

Бу ерда “+”, “x”,”(“ ва “)” терминал белгилар бўлиб, { ва } метабелгилардир. А дан туғилувчи тил x операндли ифодалардан, “+” амал ишорасида ва қавслардан ташкил топади.

Гапларга мисоллар.

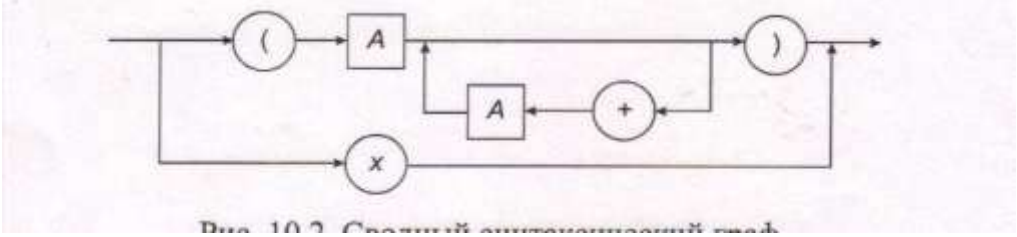
x (x) (x+x) ((x))

олтита қоидани қўлланган ҳолда олинган графлар қуйидаги 6-расмда келтирилган.



6- расм. Синтаксис графлар.

Ушбу графлар тизимини C, B ва B ни A га мос равишда алмаштириб битта графга (7-расм) келтириш мумкин.



а. 7-расм. Келтириб чиқарилган синтаксис граф.

Синтаксис граф тил грамматикасини эквивалент тасвирлаш ҳисобланади, ундан Бэкус Наур Формаси (БНФ) қоидалари ўрнига фойдаланиш мумкин. Бу жуда қулай кўриниш бўлиб, баъзи ҳолларда БНФ сидан ҳам қулайрокдир.

Граф тилни ишлаб чиқарувчиси учун таянч нуқта сифатида хизмат қилиб, жуда қулай тасвирлаш ҳисобланади.

Граф тил структураси ҳақида тўлиқ ва аниқ тассавур беради ва грамматик таҳлил жараёнини яхшироқ тассавур қилишга имкон яратади.

Битта белгини аввалдан қуришни таъминлайдиган детерминирланган грамматик таҳлилни таъминлаш учун LL(1) чегаралар ўрнатилган. Синтаксисни граф кўринишида ифодалашда улар қуйидагича ифодаланади:

1. Ҳар бир тармоқланишда шундай шоҳни танлаш керакки, у бўйича навбатдаги белги бўйича шу шоҳда таҳлил кетсин. Бу иккита бир хил шоҳ битта бир хил белгидан бошланмаслиги кераклигини билдиради.
2. Агар қандайдир А графни ҳеч қандай кирувчи белгини ўқимасдан ўтиш мумкин бўлса, у ҳолда бундай «ноллик шоҳ» барча А дан кейин келадиган белгилар билан белгиланиши керак. (Бу шу шоҳга ўтиш ечимига таъсир кўрсатади).

3. Берилган синтаксис учун грамматик таҳлил дастурини қуриш.

Қандайдир тилни англовчи дастурни унинг детерминирланган синтаксис графи асосида қуриш осон. Ушбу граф дастурнинг блок – чизмасини акс эттиради, уни ишлаб чиқаришда айлантириш қоидаларига яъни улар ёрдамида БНФдан синтаксиснинг граф тасаввурини олиш мумкин бўлган қоидаларга қаттик амал қилиш талаб қилинади. Ушбу қоидаларнинг қўлланилиши турли мақсадларга мос келувчи процедураларни ўзида жамловчи асосий дастурнинг мавжуд бўлишини таъкидлайди.

Соддалик учун таҳлил қилишимиз керак бўлган гап input кирувчи файл билан берилган ва терминал белгилар эса char туридаги алоҳида қийматлардир. Фараз қилайлик char ch ўзгарувчи ҳар доим навбатдаги ўқилиши керак бўлган белгини сақлайди. У ҳолда кейинги белгига ўтиш қуйидаги оператор орқали ифодаланади:

```
ch= fgetc(input);
```

Шуни таъкидлашимиз керакки, функция char fgetc(FILE *fp) СИ дастурлаш тилининг стандарт функцияси бўлиб, файлдан белгини ўқиш учун фойдаланилади ва ундан фойдаланиш учун дастурнинг бошланишида мос сарлавҳа файл #include <stdio.h> файл директивани жойлаштириш зарур.

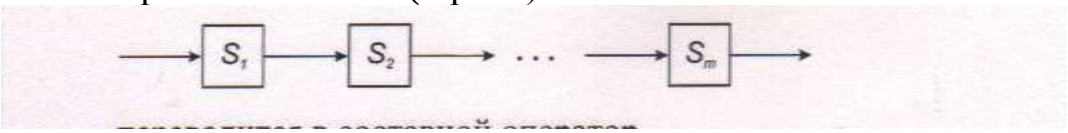
Асосий дастур биринчи белгини ўқиш учун ўқиш операторидан ташкил топади ва ундан кейин грамматик таҳлилнинг асосий мақсадини активлаштирадиган активлик оператори келади. Алоҳида грамматик

таҳлилнинг мақсадларига ёки графларига мос келувчи процедуралар қуйидаги қоидалар асосида олинади. S графни айлантириш ёрдамида олинган оператор T(S) билан белгилансин. Графни дастурга айлантириш қоидаси:

B1. Мос ўрнига қўйишлар орқали графлар тизимини мумкин қадар кичик сонли алоҳида графларга келтириш.

B2. B3-B7 пастда келтирилган қоидалар ёрдамида ҳар бир графни процедура ифодасига айлантириш.

B3. Элементлар кетма-кетлиги (8-расм)

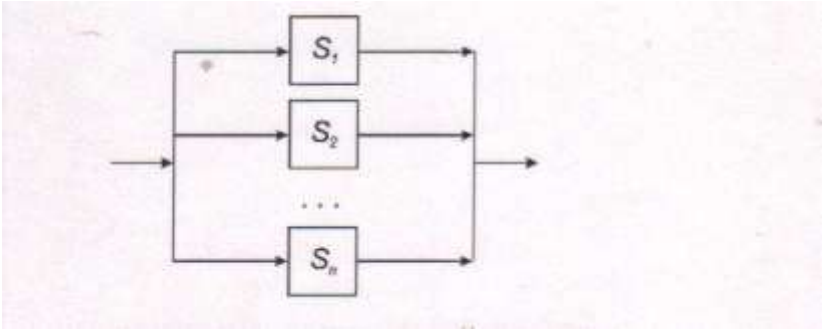


8-расм.

таркибли операторга ўтказилади.

```
{
T(S1);
T(S2);
……;
T(Sn)
}
```

B4. Элементлар танлови (9-расм)



9-**расм.**

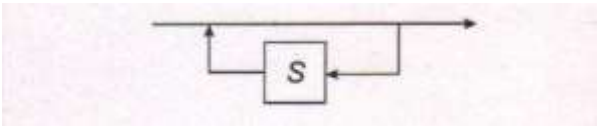
шартли операторга ўгирилади

```
if(belongsTo (ch, L1)) T(S1);
        else if(belongsTo (ch, L2)) T(S2);
else ….
                                        if(belongsTo (ch, Ln)) T(Sn);
else error();
```

бу ерда Li Si(Li=first(Si)) конструкциянинг бошланғич белгилар тўпламини билдиради, belongsTo (ch,L1) функция эса хақиқий қийматни қайтаради, агар

ch белги Li бошлангич белгилар тўпламига тегишли бўлса, ва ёлғон қиймат қайтаради акс ҳолда.

B5. Қуйидаги кўринишдаги такрорлаш (10-расм)



10-расм.

қуйидаги операторга ўгирилади

while(belongsTo (ch, L)) T(S);

бу ерда T(S) B3-B7 қоидаларга мос равишда S ни акс эттирилиши, а L эса бу L=first(S) тўпламдир.

B6. Бошқа A графни белгиловчи элементни (11-расм)

11-расм.

A() функцияга мурожат қилувчи операторга ўгирилади.

B7. x терминал белгини белгиловчи граф элементи (12-расм)

12-расм.

қуйидаги операторга ўгирилади.

if(ch = = ‘x’) ch = fgetc(input);

else error();

бу ерда error() функцияга нотўғри конструкцияга дуч келинганда мурожаат этилади.

Энди ушбу қоидаларни редуцив графни грамматик таҳлил дастурига айлантириш мисолида кўриб чиқамиз.

```
  char ch;
  void A( )
 {
  if( ch= = ‘x’) ch= fgetc(input);
  else
    if(ch= =’(‘)
```

```
        {
    ch= fgetc(input);
      A( );
    while(ch= =’+’)
            {
    ch= fgetc(input);
A( );
              }
  if( ch= = ‘)’) ch= fgetc(input);
  else error( );
            }
    else error( );
     }
void main(int argc. char **argv)
{
  ch= fgetc(input);
      A( );
 }
```

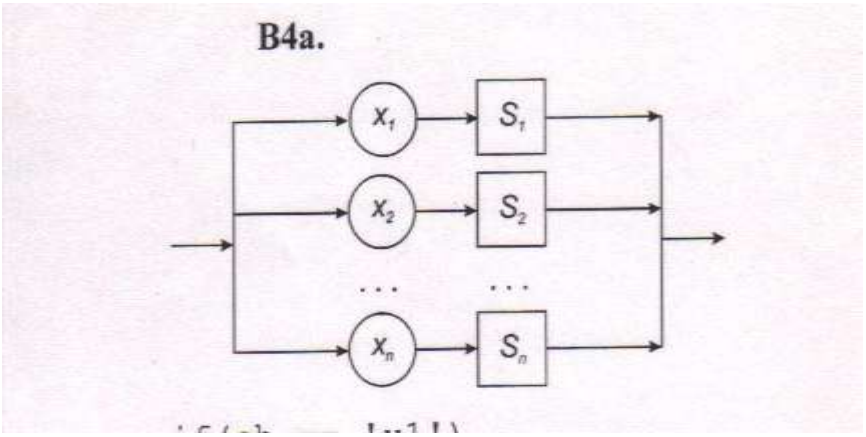
Ушбу айлантиришда дастурлашнинг бир неча дастурни соддалаштирувчи қоидалари қўлланилди. Масалан, бирга бир айлантиришда тўртинчи қатор қуйидаги кўринишга эга бўлар эди.

```
  if( ch= = ‘x’)
    if( ch= = ‘x’) ch= fgetc(input); else error( );
  else….
```

Ушбу ҳолни худди шу дастурда кўриб ўтилгандек соддалаштириш мумкин. Саккизинчи ва ўн иккинчи қатордаги ўқиш операторлари ҳам худди шундай соддалаштириш асосида олинган.

Бундай соддалаштиришлар мумкин ёки йўқлигини аниқлаш ва уларни графлар кўринишида кўрсатиш фойдалироқдир.

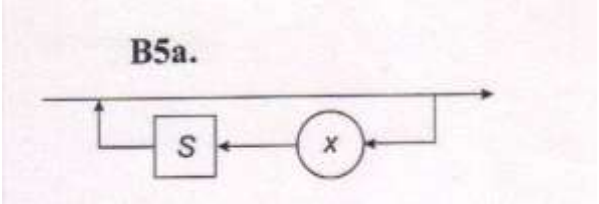
Иккита асосий ҳолат қуйидаги қўшимча қоидалар билан ёпилади:
B4a. (13-расм)



13-расм.

```
if( ch= = ‘x1’)
     {
   ch= fgetc(input);
      T(S1);
     }
   else
   if( ch= = ‘x2’)
       {
   ch= fgetc(input);
      T(S2);
        }
   else
        ….
   if( ch= = ‘xn’)
           {
 ch= fgetc(input);
 T(Sn);
           }
   else error( );
```

B5a.(14-расм)



14-расм.

```
while(ch= =’x’)
              {
ch= fgetc(input); T(S );
```

```
                }
```
Бундан ташқари кўпинча учрайдиган қуйидаги конструкцияни

```
ch= fgetc(input); T(S );
while(W)
                {
   ch= fgetc(input); T(S );
                 }
```

қисқачароқ қуйидагича ифодалаш мумкин:

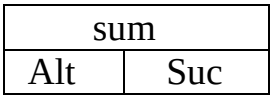
```
do {
        ch= fgetc(input);
        T(S );
}  while(W);
```

Биз бу ерда error («хатолик») функциясини ифодаламаймиз, чунки бизни ҳозирча кирувчи гапни тўғри эканлигини аниқлаш қизиқтиради холос.

4. Грамматик тахлилнинг жадвалли-бошқарув дастурини қуриш.

Жадвалли –бошқарилувчи универсал дастурда аниқ грамматикалар таҳлил қилиниши керак бўлган гапларда кирувчи берилганлар кўринишида берилади. Универсал дастур грамматик таҳлилнинг оддий пастдан чиқувчи усулига мос равишда ишлайди, шу сабабли, агар детерминирланган синтаксис графга асосланган бўлса, у соддадир, яъни гапни қайтишсиз битта белгига аввалдан қараб таҳлил этиш мумкин.

Берилган ҳолда грамматика дастурнинг структурасига эмас, балки берилганларнинг мос келувчи структурасига айлантирилади. Графни тасвирлашнинг табиий усули -бу ҳар бир белги учун тугун киритиш ва ушбу тугунларни кҳрсаткичлар орқали боғлашдир. Бундан кўринадики, «жадвал» бу оддий тўплам эмас. Ушбу структуранинг тугунлари бирлашмаларга (union) бирлаштирилган структуралардан (struct) ташкил топади. Бирлашма тугуннинг тоифасини идентификациялайди. Биринчиси терминал белги билан идентификацияланади ва tsum билан белгиланади, иккинчиси берилганлар структурасига кўрсаткич бўлиб нотерминал nsum га мос келади. Бирлашманинг иккала варианти иккита кўрсаткичга эга: биттаси кейинги белгига кўрсатади, кўрсаткич бўйича (suc) , иккинчиси эса мумкин бўлган альтернативлар рўйхати билан боғлиқ (alt). Граф кўринишида тугунни қуйидагича (15-расм) ифодалаш мумкин.



15-**расм.**

Худди шундай яна бўш кетма-кетликни ифодаловчи элемент керак бўлади, «бўш» белги**.** Уни қуйидаги empty терминал элемент ёрдамида белгилаймиз.

```
struct node;
```

```
tupedef node *pointer;
struct node {
   pointer suc;
   pointer alt;
   int isTerminal;
  union {
   char tsum;
   hpointer nsum;
        };
           };
```

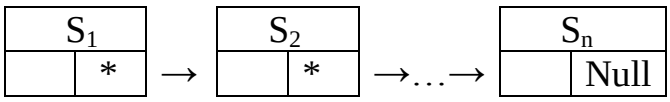
Графни берилганлар структурасига айлантириш қоидалари B1-B7 қоидалар билан бир хил.

Графни берилганлар структурасига айлантириш қоидалари:

C1. Мос ўрнига қўйишлар ёрдамида графлар тизимини мумкин қадар кам сонли алоҳида графларга келтириш.

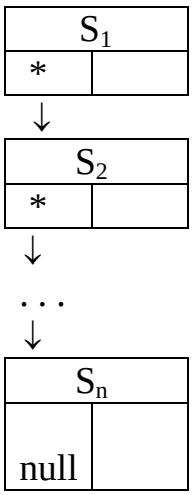
C2. Қуйида келтирилган C3-C5 қоидаларга мос равишда ҳар бир графни берилганлар структурасига айлантириш.

C3. B3 қоидадаги элементлар кетма-кетлиги қуйидаги (16-расм) тугунлар рўйхатига айлантирилади:



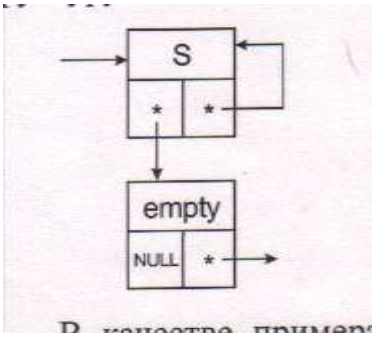
16-расм.

C4. B4 қоидадаги альтернативлар рўйхати қуйидаги (17-расм) берилганлар структурасига айлантирилади.



17-расм.

C5. B5 қоидадаги такрорлашлар (цикл) қуйидаги (18-расм) структурага айлантирилади.



18-расм.

Мисол сифатида юқоридаги 7-расмда келтирилган келтириб чиқарилган синтаксис граф учун берилганлар структурасини қурамиз.

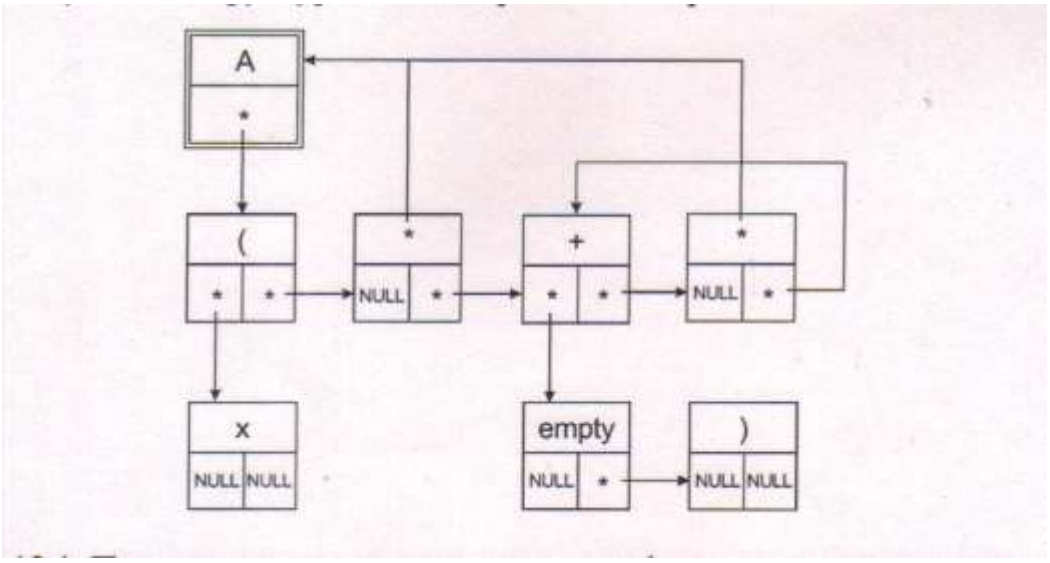
Берилганлар структураси шу структура тегишли бўлган нотерминал белги (мақсадни) исми **<u>сарлавҳа-тугун билан</u>** идентификацияланади. Умуман олганда сарлавҳа зарур эмас, балки унинг ўрнига мақсад майдони ўрнига мос структурага «кириш» ни кўрсатиш мумкин. Лекин сарлавҳадан чоп этилаётган структура исмини сақлаш учун фойдаланиш мумкин:

```
struct header;
tupedef header * hpointer;
struct header {
  pointer entry;
    char sym;
          };
```

Кирувчи файлда белгилар кетма-кетлиги кўринишида тасвирланган гапни грамматик таҳлилини амалга ошираётган дастур бир тугундан кейинги тугунга ўтишни ифодаловчи такрорланувчи операторлардан ташкил топади.

Терминал белги учун ажратилган sum майдонига белгини ўзи жойланади, нотерминал белги учун эса мос берилганлар структурасига кўрсаткич жойлашади. У граф интерпретациясини берувчи процедура каби ифодаланади, агар нотерминал белгини ифодаловчи тугун учраб қолса, у ҳолда ушбу берилган тугун жорий графни интерпретациясини тугашига олиб келади. Шундай қилиб интерпретация процедураси рекурсив чақирилади. Агар кирувчи файл жорий белгиси (sum) берилганлар структурасининг жорий тугундаги белгиси билан мос тушса, у ҳолда процедура suc майдони кўрсатаётган тугунга ўтади, акс холда alt майдони кўрсатаётган тугунга ўтади.

Олинган структура 19- расмда келтирилган.



19-расм. Синтаксис графни берилганлар структураси кўринишида ифодалаш.

```
void parse (hpointer goal, int &match)
{
  pointer s;
   s= goal       entry;
   do
     {
     if (s     isTerminal)
       {
     if(s     tsym  = = sym)
         {
   match= 1;
   sym = fgetc(input);
          }
     else
     match =(s     tsym   = =  empty )? 1: 0;
          }
     else
   parse(s        nsym, match);
   if(match)
      s= s    suc;
      else
                s=  s     alt;
             }   while(s!=NULL);
           }
```

Грамматик таҳлилнинг бундай ташкил этилишида дастур кўпинча тилнинг аниқ бир форматда берилган грамматикасини ўқийди ва берилганларнинг мос структураларини тўлдиради ва фақат шундан кейингина берилган тилдаги матнни ўқийди ва уни синтаксис таҳлил қилади.

Назорат саволлари.

1. Синтаксис граф нима?
2. Синтаксис графнинг қуриш қоидалари хақида маълумот беринг.
3. Синтаксис графни қуришдан мақсад нима?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза №12.

Мавзу: Юкориловчи синтаксис тахлил.

Режа:

1. **«Перенос-свертка» туридаги синтаксис тахлил.**
2. **Белгилар жадвали билан ишлаш.**
3. **LL(1) –грамматикалар.**
4. **Грамматикаларни LL(1) кўринишга айлантириш.**

Калит сузлар.

- **ПС-тахлил**
- **Барг**
- **Илдиз**
- **Қатор**
- **Махсулот**
- **Қисм қатор**
- **S-грамматика**
- **Йўналтирувчи белгилар**
- **Хамрох-белги**
- **Рекурсия**
- **Факторлаш**

1.«Перенос-свертка» туридаги синтаксис тахлил.

Ушбу булимда юкориловчи синтаксис тахлилнинг « перенос-свертка» туридаги синтаксис тахлилнинг асосий усуллари каралади. Бундан буёгига кискача ПС-тахлил деб аталади.

ПС-тахлил барглардан бошлаб ва илдиздан дарахт юкорисига караб ишлайди ва кирувчи каторнинг разбор дарахтини куришга харакат килади. Бу жараённи w каторни грамматиканинг бошлангич белгисига свертка-алмашиш сифатида караш мумкин. Алмашувнинг хар бир кадамида кандайдир махсулотнинг унг булакидаги кисмкатор махсулотнинг чап булагидаги белги билан алмаштирилади ва агар хар бир кадамда кисмкаторлар тугри ва аник танланса, у холда биз унгдан келтириб чикарилувчи мурожатни оламиз.

Мисол:

Куйидаги грамматикани караймиз.

S	aABe
A	Abc| b
B	d

abbcde гап S га куйидаги кадамлар ёрдамида келтирилади:

abbcde

aAbcde
aA de
aABe
S

Биз кандайдир махсулотни унг булагига мос келувчи abbcde каторни кисмкаторни кидириш максадида сканерлаймиз. Бундай кисмкаторлар булиб b ва d хисобланади. Энг четки чапки b ни оламиз ва уни A b; махсулотнинг чап булаги булган нотерминал A билан алмаштирамиз., шундай килиб aAbcde каторни оламиз. Энди махсулотнинг унг булагига Abc, b ва d кисмкаторлар мос келади. махсулотнинг унг булагига мос келувчи b энг четки чап кисмкатор булса хам кисмкаторни алмаштириш учун Abc кисмкаторни танлаймиз ва уни A Abc махсулотга мос нотерминал A билан алмаштирамиз. Натижада aAde каторни оламиз. Махсулотнинг чап булагини B d да dни B га алмаштириб aABe ни оламиз ва у биринчи махсулотга мос равишда бошлангич белги S билан алмаштирилади. Шундай килиб, туртта алмаштиришдан келиб чикадиган кетма-кетлик abbcde каторни бошлангич белги S га келтиришга имкон беради. Бу кискартиришлар унгдан келтириладиган алмаштиришлар (яъни тескари тартибда ёзилган) ни ифодалайди.

S $\rightarrow$ aABe $\rightarrow$ aA de $\rightarrow$ aAbcde $\rightarrow$ abbcde

2. Белгилар жадвали билан ишлаш.

Синтаксис тахлилчи купинча контекст-озод грамматикадан фойдаланганлиги сабабли, тилнинг контекст-озод булакларини аниклаш усулини топишимиз керак. Масалан, купгина тилларда идентификаторлардан уларни ифодаламагунча фойдаланиб булмайди, худди шундай яна дастурда турли тоифадаги кийматлардан фойдаланишга хам чегаралар мавжуд. Ифодаланган идентификаторларни ва уларнинг турларини саклаб туриш учун купгина компиляторларда белгилар жадвалидан фойдаланилади.

Идентификатор ифодаланаётганда, масалан int a; a ни аникловчи амалга оширилиши деб аталади. Лекин, a бошка контекстда хам учраши мумкин: a=4 ёки a+в ёки read(a) бу ерда a нинг амалий амалга оширилиши назарга тутилаяпти.

Идентификаторни аникловчи амалга оширилишида компилятор объектни белгилар жадвалига жойлайди. Амалга оширилувчи холда эса белгилар жадвалида мос аникловчи объектни элементини, компиляция вактида талаб килинган уни турини ва бошка белгиларини билиш учун, кидирилади.

Купгина дастурлаш тилларида бита идентификатордан дастурнинг турли булакларида турли объектларни ифодалаш учун фойдаланилади. Бундай холларда дастурнинг структураси ушбу объектларни фарклашга ёрдам беради, масалан

{
int a;
…..

```
}
…..
{
char a;
…..
}
```

Ушбу холда а иккита турли блокда иккита турли объектни ифодалайди. Белгилар жадвали битта идентификатордан икки марта фойдаланишда идентификаторларни фарклаш учун худди дастурдаги блокли структурага эга булиши керак.

Купгина замонавий юкори даража тиллари куйидаги хусусиятларга эга булиши керак:

Идентификаторни аникловчи амалга оширилиши амалга оширилишидан кура матнда вактлирок пайдо булиши керак.

Идентификаторни амалий ишлатилишида мос келувчи аникловчи амалга оширилиш ушбу идентификаторни ифодалаш сакланган блокда жойлашади.

Битта блокда идентификатор бир мартадан ортик ифодаланиши мумкин эмас.

Компиляторнинг белгилар жадвалида идентификатор хакидаги, компиляция жараёнида керак буладиган, бошка маълумотлар хам сакланиши мумкин. Масалан, идентификаторнинг манзили, ёки костантанинг киймати.

Дастур ичида белгилар жадвалининг амалга оширилиши туплам ёки структура занжири куринишида булиши мумкин. Бу ерда хар бир элемент узидан кейинги, балки узидан олдинги элементга хам курсатади.

Туплам куринишидаги реализация.

Бундай ташкил этишнинг ютуклари:

Белгилар жадвалига тезликда хотира ажратилиши (ьир марта тупламни эълон этилиши пайтида амалга оширилади).

Тупламнинг бошка элементларини жойлаштириш хакидаги маълумотни саклаш зарурияти йуклиги сабабли, жойни иктисод килишга эришилади.

Тезкор кидирув усулларини амалга оширишнинг соддалигига эришилади (стекни имитацияси ва х.к.).

Камчиликлари:

Кичкина дастурлар ва кам идентификаторлар иштирок этган холдаги трансляция жараёнида хотиранинг фойдасиз ишлатилиши, чунки тупламнинг каттагина кисми ишлатилмай колиб кетади.

Катта дастурларни трансляциясида вактида шундай холат юзага келиши мумкинки, узгарувчилар хакидаги маълумотни жойлаштириш учун физик хотира бор, лекин туплам эса тулиб булган. Бу холда мос равишда тулиб кетиш сабабли трансляцияни амалга ошириб булмайди.

Занжирли структура куринишидаги реализация (богланган руйхат).

Бундай ташкил этишнинг **ютукларидан** бири бу хотира ресурсларидан тулик фойдаланишдир.

Камчиликлари:

1. Хотира ажратилиши нисбатан секин амалга оширилади, чунки хар бир элемент учун алохида ажратилиди.

2. Хотиранинг кушимча харажатланиши, яъни кейинги ва олдинги элементга курсаткичлар сакланади.

Амалиётда яна ушбу куринишларнинг комбинацияси хам учраши мумкин.

Ифодаланиш учраши билан белгилар жадвалининг мос элементи стекнинг юкори кисмига жойлашади, блокдан чикишда эса, ушбу блокдаги ифодалашга мос келувчи, белгилар жадвалининг барча элементлари стекдан кетадилар. Стекнинг курсаткичи блокка кириш пайтидаги холатигача пасаяди. Натижада разборнинг ихтиёрий вактида белгилар жадвалининг жорий идентификаторларга мос келувчи элементлари стекда жойлашади, улар билан боглик идентификаторларни амалий ва аникловчи реализациялари стекда юкоридан пастга кидирувни талаб килади. Занжирли структура урнига стекдан фойдаланиш занжир структурасида курсаткичлар банд этган жойни иктисод килиш имконини беради.

Караб чикилган усул куйидаги расмда келтирилган.

Дастур куриниши	Стек
{	d char
int a,b;	c char
{	b int
char c, d;	a int
}	f int
{	e int
int e,f;	b int
}	a int
}	

Расм Стек билан амалга оширилиш варианти.

3..LL(1)–грамматикалар.

LL(1) –грамматика – бу шундай грамматикаки, унинг асосида юқоридан пастга тамойили билан ишловчи детерминирланган синтаксис таҳлилчини олиш мумкин. LL(1) –грамматикага аниқ таъриф беришдан аввал s –грамматика тушунчасини киритамиз.

s –грамматика бу шундай грамматикаки, унда:

1) ҳар бир туғилувчи қоиданинг ўнг қисми терминалдан бошланади.

2) чап бўлакда биттадан кўп туғилувчи қоидалар бўлган ҳолида биттагина нотерминал пайдо бўлади, мос ўнг бўлаклар турли терминаллардан бошланади.

Биринчи шарт, грамматика Грейбахнинг нормал формасига эга, фақат ҳар бир ўнг бўлак қоидасининг бошланишида терминалдан кейин нотерминаллар ёки терминаллар келиши мумкин деган таъкидга ўхшашдир.

Иккинчи шарт детерминирланган пастки таҳлилчини ёзиш имконини беради, чунки тилнинг гапларини чиқаришда ҳар доим альтернатив туғилувчи қоидалар ўртасида энг чап нотерминал учун сентенциал формада, аввалдан битта кейинги белгини текшириб, танлов қилиш мумкин.

Қуйидаги туғилувчи қоидалар билан аниқланган грамматика

$S \rightarrow pX$ $X \rightarrow x$
$S \rightarrow qY$ $Y \rightarrow y$
$X \rightarrow aXb$ $Y \rightarrow aYd$

s –грамматикани ташкил этади, у ҳолда қуйидаги грамматика эса, худди ўша тилни генерация қилади ва у s –грамматика эмас.

$S \rightarrow R$ $X \rightarrow aXb$
$S \rightarrow T$ $X \rightarrow x$
$R \rightarrow pX$ $Y \rightarrow aYd$
$T \rightarrow qY$ $Y \rightarrow y$

чунки иккита қоиданинг ўнг бўлаклари терминаллардан бошланмайди.

s–грамматикадан берилган грамматика сифатида фойдаланилаяптими ёки йўқми аниқлаш жуда осон, баъзи ҳолларда s –грамматика бўлмаган грамматикани ҳам s –грамматикага генерация қилинган тилни безовта қилмаган ҳолда айлантириш мумкин.

paaaxbbb қаторни таҳлилини юқоридаги s–грамматикадан фойдаланилган ҳолда қараймиз. S белгидан бошлаб қаторни генерация қилишга бошлаймиз. Чап томонли чиқишни қўллаймиз. Натижа эса қуйидагича бўлади.

Бошланғич қатор: paaaxbbb

Чиқиш: $S \rightarrow pX \rightarrow paXb \rightarrow paaXbb \rightarrow paaaXbbb \rightarrow paaaxbbb$

Чиқишда сентенциал кўринишдаги бошланғич терминаллар бошланғич қатордаги белгилар билан солиштирилади. Сентенциал кўринишдаги энг чап терминалларни биттадан кўп туғилувчи қоидаларни қўллаб алмаштириш мумкин бўлган ҳолларда, ҳар доим мос туғилувчи қоидани кирувчи қатордаги кейинги белгина текшириб танлаш мумкин. Бу шу билан боғлиқки, биз s –грамматикадан фойдаланганлигимиз туфайли альтернатив туғилувчи қоидаларнинг ўнг қисмлари турли терминаллардан бошланади. Шундай қилиб ҳар доим юқоридан пастга таҳлилни амалга оширувчи детерминирланган таҳлилчини s –грамматикадан генерация қилинадиган тил учун ёзиш мумкин.

LL(1) –грамматика s –грамматикани умумлаштирилгани ҳисобланиб, унинг умумлаштириш тамойили пастдан чиқувчи детерминирланган таҳлилчиларни қуриш имконини беради. Иккита LL ҳарфи қаторлар чапдан ўнгга (Left) қаралаётганлигини ва энг чап чиқишлардан (Left) фойдаланилишини билдиради, 1 рақами эса туғилувчи қоидаларнинг вариантлари аввалдан қурилган битта белги ёрдамида танланишини англатади. Демак, LL(2) – грамматика ҳақида гап кетса, бу қаторлар чапдан ўнгга қараб қаралишини ва энг чап чиқишлардан фойдаланилишини, ва туғилувчи қоидаларнинг вариантлари иккита аввалдан қурилган белгилар ёрдамида танланишини англатади.

Худди шунингдек, LR(1) –грамматика қаторлар чапдан ўнгга (Left) қаралаётганлигини ва энг ўнг чиқишлардан (Right) фойдаланилаётганлигини билдиради, туғилувчи қоидалар вариантлари эса аввалдан қурилган битта белги ёрдамида танланади.

Агар туғилувчи қоиданинг ўнг қисми терминалдан бошланмаган ҳолда ҳам нотерминал белгининг ҳар бир варианти аниқ бир терминаллар тўпламидан бошланган қаторларга бошланиш бериши мумкин.

Масалан қуйидаги туғилувчи қоидалар асосида

$S \to RY$ $R \to paXb$

$S \to TZ$ $T \to qaYd$

келтириб чиқариш мумкинки, R ва T учун бошқа туғилувчи қоидалар бўлмасалар $S \to RY$ ни юқоридан пастга (пастга томон таҳлил- бошланғич белгидан қаторгача) таҳлилда қўллаш, фақатгана аввалдан қараб чиқилган белги p бўлган ҳолда мақсадга мувофиқдир.

Худди шундай туғилувчи қоида $S \to TZ$ дан бундай белги q бўлган ҳолда фойдаланиш тавсия этилади.

Бу эса қуйидаги ҳамрох белгилар тўплами ғоясини келтириб чиқаради:

$a \in S(A) \Leftrightarrow A \Rightarrow a\alpha$,

бу ерда A – нотерминал белги, α – терминаллар қатори ва/ёки нотерминаллар қатори, бўш бўлиши мумкин, S(A) эса A ҳамрох белгилар тўпламини англатади. Қуйидаги туғилувчи қоидаларга эга грамматикада

$P \to Ac$ $A \to aA$

$P \to Bd$ $B \to b$

$A \to a$ $B \to bB$

a ва b белгилар P учун ҳамрох-белгилардир.

Худди шундай

$a \in S(\alpha) \Leftrightarrow \alpha \Rightarrow a\beta$ қатор терминал ва/ёки нотерминаллар учун ҳам ҳамрох –белгиларни аниқлашимиз мумкин.

Бу ерда α ва β терминаллар ва/ёки нотерминалларнинг қаторлари (β бўш қатор ҳам бўлиши мумкин).

Грамматика LL(1) хусусиятларига эга бўлишининг зарурий шарти шуки, ўнг томон альтернатив туғилувчи қоидаларининг ҳамрох белгилари тўпламлари бир-бирлари билан кесишмаслигидир.

Ўнг томоннинг бошланишида нотерминал белги бўш қаторни генерациялаётган вақтда эҳтиёткорлик зарур. Масалан, қуйидаги грамматикада

$P \to AB$	$A \to \varepsilon$
$P \to BG$	$B \to bB$
$A \to aA$	$B \to c$

$S(AB)= \{a,b,c\}$, га эга бўламиз ва бу ерда b ва c тўпламга киради, чунки A бўш қаторни генерация қилиши мумкин; $S(BG)= \{b,c\}$ – тўпламлар кесишадилар ва бундан кўринадики ушбу грамматика LL(1) грамматика эмас.

Яна қуйидаги туғилувчи қоидаларга асосланган грамматикани қараймиз:

$T \to AB$	$Q \to \varepsilon$
$A \to PQ$	$B \to bB$
$A \to BC$	$B \to e$
$P \to pP$	$C \to cC$
$P \to \varepsilon$	$C \to f$
$Q \to qQ$	

Ушбу грамматика учун $S(PQ) = \{p, q\}$ ва $S(BC)=\{b,e\}$.

Лекин PQ бўш қаторни генерациялаши мумкин бўлгани учун кейинги қараладиган белги $A \to PQ$ туғилувчи қоидани b ёки e бўлиши мумкин.

Шунинг учун йўналтирувчи белгилар тушунчаси киритилади, ва у қуйидагича аниқланади: агар A нотерминал бўлса, у ҳолда унинг йўналтирувчи белгилари бўлиб $S(A)+$ (A дан кейинги барча белгилар, агар A бўш қаторни генерация қилиш мумкин бўлса).

Бошқача айтганда, бу барча ҳамрох-белгилар тўплами + A дан кейинги барча белгилар, агар A бўш қаторни генерация қилиши мумкин бўлса. Умумий ҳолда a нотерминалнинг $P(P \to a)$ берилган варианти учун қуйидагига эга бўламиз:

$$DS(P, a)=\{a \mid a \in S(a) \text{ ёки } (a \Rightarrow \varepsilon \text{ ва } a \in F(P))\}$$

Бу ерда F(P) P дан кейин келувчи белгилар тўпламидир. Юқорида келтирилган мисолда йўналтирувчи белгилар -бу қуйидагилардир:

$DS(A,PQ)= \{p,q,b,e\}$
$DS(A,BC)=\{b,e\}$

Кўрсатилган тўпламлар кесишганлиги сабабли, ушбу грамматика пастга томон детерминирланган таҳлилчи учун асос бўла олмайди.

LL(1) грамматика таърифини аниқлаштирамиз: Грамматикани LL(1) грамматика деб атаймиз, агар биттадан кўп туғилувчи қоидаларда учрайдиган ҳар бир нотерминал учун, альтернатив туғилувчи қоидаларнинг ўнг қисмига мос йўналтирувчи белгилар тўплами кесишмасалар. Барча LL(1) грамматикаларни юқоридан пастга детерминирланган ҳолда таҳлил қилиш мумкин.

4. Грамматикаларни LL(1) кўринишга айлантириш.

Кўпгина дастурлаш тиллари учун «кўриниб турган» грамматика бу LL(1) грамматика эмас. Лекин, кўпинча дастурлаш тилининг жуда катта сонли контекст-озод воситаларини LL(1) грамматика орқали ифодалаш мумкин. Муаммо шундаки, LL(1) грамматика белгиларига эга бўлмаган грамматикага эга бўлган ҳолда, унга тенг кучли бўлган LL(1) грамматикани топиш керак. Ихтиёрий контекст-озод грамматикани LL(1) кўринишга айлантиришнинг умумий алгоритми мавжуд эмас. Лекин кўпгина хусусий ҳолларда бундай айлантиришларни бажарувчи қатор усуллар мавжуд.

Чап рекурсияни йўқотиш.

Чап рекурсияга эга грамматика LL(1) грамматика ҳисобланмайди. Қуйидаги қоидаларни кўриб чиқамиз:

$A \rightarrow Aa$ (A да чап рекурсия)

$A \rightarrow a$

Бу ерда a белги-ҳамрох A нотерминалнинг иккала варианти учун.

Худди шунингдек чап рекурсив циклга эга грамматика ҳам LL(1) грамматика ҳисобланмайди, масалан

$A \rightarrow BC$

$B \rightarrow CД$

$C \rightarrow AE$

Чап рекурсив циклга эга грамматикани фақат тўғри чап рекурсияга эга грамматикага айлантириш мумкин ва кейинчалик қўшимча нотерминаллар киритиш йўли билан чап рекурсияни тўлиқлигича олиб ташлаш мумкин бўлади (ҳақиқатда эса у ўнг рекурсия билан алмаштирилади. Ўнг рекурсия эса LL(1)- хусусиятларига нисбатан муаммо ҳосил қилмайди).

Мисол сифатида қуйидаги туғилувчи қоидаларга ва чап рекурсив циклга эга бўлиб A,B,C,D ларни ўзига олувчи грамматикани қараб чиқамиз:

$S \rightarrow Aa$ $\quad$ $C \rightarrow Dd$

$A \rightarrow Bb$ $\quad$ $C \rightarrow e$

$B \rightarrow Cc$ $\quad$ $D \rightarrow Az$

Ушбу циклни тўғри чап рекурсияга ўзгартириш учун нотерминалларни қуйидагича тартибга соламиз: S, A,B,C,D.

Барча қуйидаги кўринишдаги туғилувчи қоидаларни қараймиз:

$Xi \rightarrow Xjy$,

Бу ерда Xi ва Xj нотерминаллар, у эса терминал ва нотерминал белгилар қатори. j >=i бўлган қоидалар учун ҳеч қандай харакат бажарилмайди. Лекин бу тенгсизлик чап рекурсив циклга эга барча қоидалар учун ҳам мос келавермайди. Юқоридаги биз танлаган тартибда қуйидаги ягона қоида билан ишлаймиз.

$D \rightarrow Az$

чунки бу тартибда A D дан кейин келмоқда. Энди A ни чап томондаги барча мавжуд қоидалардан фойдаланиб алмаштирамиз. Натижада

$D \rightarrow Bbz$ ни оламиз.

Тартибда B D дан кейин келаётганлиги сабабли, жараён такрорланади, буни эса қуйидаги қоида беради:

$D \rightarrow Ccbz$

Сунгра у яна такрорланади ва иккита қоидани беради:

$D \rightarrow ecbz$

$D \rightarrow Ddcbz$

Энди келтириб чиқарилган грамматика қуйидагича бўлади:

$S \rightarrow Aa$	$C \rightarrow e$
$A \rightarrow Bb$	$D \rightarrow Ddcbz$
$B \rightarrow Cc$	$D \rightarrow ecbz$
$C \rightarrow Dd$	

Барча ушбу туғилувчи қоидалар талаб қилинган кўринишга эга, чап рекурсив цикл эса тўғри чап рекурсия билан алмаштирилган. Тўғри чап рекурсияни йўқотиш учун янги нотерминал белги Z киритамиз ва қоидаларни алмаштирамиз:

$D \rightarrow ecbz$

$D \rightarrow Ddcbz$

ларни

$D \rightarrow ecbz$	$Z \rightarrow dcbz$
$D \rightarrow ecbzZ$	$Z \rightarrow dcbzZ$

D айлантиришгача ва ундан сўнг қуйидаги регуляр ифодани генерациялайди

(ecbz) (dcbz)*

Умумлаштирган ҳолда, шуни кўрсатиш мумкинки, агар A нотерминал r+s туғилувчи қоидаларнинг чап қисмида учраса, r булардан тўғри чап рекурсиядан фойдаланса, s эса фойдаланмаса, яъни

$A \rightarrow A\alpha_1, A \rightarrow A\alpha_2, \ldots, A \rightarrow A\alpha_r$

$A \rightarrow \beta_1, A \rightarrow \beta_2, \ldots, A \rightarrow \beta_s$

у ҳолда бу қоидаларни қуйидагиларга алмаштириш мумкин:

$$\left.\begin{array}{l} A \rightarrow \beta i \\ A \rightarrow \beta iZ \end{array}\right\} \; L \le i \le s \qquad \left.\begin{array}{l} Z \rightarrow \alpha i \\ Z \rightarrow \alpha iZ \end{array}\right\} \; L \le i \le r$$

Ноформал исбот шуни кўрсатадики, А айлантиришгача ва ундан сўнг қуйидаги регуляр ифодани генерациялайди:

$(\beta_1 | \beta_2 | \ldots | \beta_s)(\alpha_1 | \alpha_2 | \ldots . | \alpha_r)^*$

Шуни таъкидлаш лозимки, чап рекурсияни (ёки чап рекурсив циклни) йўқотиш натижасида LL(1) грамматикага эга бўламиз, шундай қилиб, олинган грамматика қоидаларининг чап қисмидаги баъзи бир нотерминалларга альтернатив ўнг қисмлар мавжуд бўлиб, улар бир хил белгилардан бошланадилар. Шунинг учун чап рекурсияни йўқотилгандан кейин грамматикани LL(1) кўринишгача айлантиришни давом эттириш керак.

Назорат саволлари.

1. Синтаксис тахлилда белгилар жадвалидан фойдаланиш зарурияти нимада?
2. Аникловчи реализацияни амалий реализациядан фарки нимада?
3. Белгилар жадвалининг реализациясининг туплам ва богланган руйхат куринишидаги ютук ва камчиликларини санаб беринг.
4. Нима учун чап рекурсия LL(1) –разбор учун тўсиқ ҳисобланадилар?
5. Грамматика қоидаларини факторлаш жараёни ҳақида маълумот беринг.

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза №13.

Мавзу: Синтаксис хатоликлар вактида тиклаш.

Режа:

1.Нотўғри дастур эгаси хақиқатда нималарни назарда тутганлиги хақида фикр юритиш керак.

2.Кирувчи кетма-кетликни кандайдир кисмини ўтказиб юбориш керак.

3.Матнни тиклашга уриниш ва мумкин бўлмаган холларда кетма-кетликни баъзи бўлакларини ўтказиб юбориш керак.

Калит сузлар.

- Синтаксис тахлил
- Конструкция
- Процедура
- Тиклаш белгилари тўплами
- Мақсадчалар

1.Нотўғри дастур эгаси хақиқатда нималарни назарда тутганлиги хақида фикр юритиш керак.

Дастур матнини тахлили жараёнида транслятор хатоликлар хакидаги мос диагностикани бериши ва грамматик разбор жараёнини, кейинги мумкин булган хоталикларни топиш учун, давом эттириши керак.

Ишни давом эттириш учун :

1. нотугри дастур эгаси хакикатда нималарни назарда тутганлиги хакида фикр юритиш керак.
2. кирувчи кетма-кетликни кандайдир кисмини утказиб юбориш керак.
3. матнни тиклашга уриниш ва мумкин булмаган холларда кетма-кетликни баъзи булакларини утказиб юбориш керак.

Дастурчининг дастурда нималарни назарда тутганлиги хакида тугри хулоса чикариш анчагина кийин масала, шунинг учун ушбу усул кам кулланилади.

2.Кирувчи кетма-кетликни кандайдир кисмини ўтказиб юбориш керак.

Иккинчи йуналиш буйича куйидагича курсатмалар бериш мумкин.

Оддий структурали тил холида тиклаш жараёни анчагина енгиллашади. Агар хатолик юз берган холда кирувчи кетма-кетликнинг баъзи кисмини утказиб юборилса, у холда тил албатта нотугри ишлатилиш имконияти кам булган ва грамматик разборда тиклаш учун фойдаланиш мумкин булган калит сузлар (хизматчи сузлар)дан иборат булиши керак.

Грамматик разбор дастурини куриш буйича:

Нотугри конструкция учраган вактда процедура кирувчи матнни качонки коида буйича тилнинг конструкциясидан кейин келадиган белги учрагунича утказиб юбориши керак. Бу хар бир грамматик разбор

процедурасига уни активлаштириш вактида ташки белгилар туплами (каралаётган конструкциядан кейин келаувчи) аник булиши кераклигини англатади. Хар бир процедура охирида аник уйидаги текширив куйилади: кирувчи матннинг кейинги белгиси ушбу ташки белгилар орасида мавжудми? Хатоликни топганидан сунг процедура мустакил равишда хатолик хакида маълумот бериб ва ишни тухтатмасдан матнни каердан бошлаб тахлилни тиклаш керак булган жойигача караб, чикиши керак. Шундай килиб, грамматик разборнинг процедурасидан тугри тугатилишдан бошка халокатли чикиш йуллари булмаганлиги максадга мувофикдир.

3.Матнни тиклашга уриниш ва мумкин бўлмаган холларда кетма-кетликни баъзи бўлакларини ўтказиб юбориш керак.

Дастур матнидаги катта булакларни кейинги ташки белгиларгача ташлаб кетиш яхши натижаларга олиб келмаслиги мумкин. Шунинг учун белгилар тупламига конструкцияни бошланишини белгиловчи ва ташлаб кетиш мумкин булмаган хизматчи сузларни кушиб куйилади. Шу йул билан разбор процедурасига параметрлар сифатида оддий ташки белгилар эмас, балки тиклаш белгилари узатилади. Купинча тиклаш белгилари туплами бошидан бошлаб алохида хизматчи сузлардан ташкил топади ва грамматик разборнинг максадчалари иерархияси келиши давомида максадчаларнинг ташки белгилари билан аста-секин тулдириб борилади. Юкоридаги текширувни бажариш учун умумий функцияни киритамиз ва уни test деб атаймиз. Ушбу функция учта параметрдан иборат6

Мумкин булган кейинги белгилар туплами s1; агар жорий белги унга тегишли булмаса, у холда хатолик юз беради.

Кушимча тиклаш белгилари туплами s2; уларнинг пайдо булиши бу хатоликларни англатади, лекин уларнинг баъзиларини утказиб юбориб булмайди.

n раками хатоликка бериладиб, агар функция уни топса.

```
void test(char[ ] s1, char [ ] s2, int n)
{
   If(!belongsTo(sym,s1)
  {
    error(n);
    s1= unite (s1,s2);
       while(!belongsTo(sym<s1))
          sym = fgetc(input);
     }
  }
```

Бирон бир схема барча нотугри конструкцияларнинг барчаси учун кулланилиши бирдек фойдали булмаслиги мумкин. Ихтиёрий тиклаш схемаси хам баъзи бир хато конструкцияларни кайта ишлай олмаслиги мумкин. Лекин яхши транслятор куйидаги хусусиятларга эга булиши шарт:

- хеч кандай кирувчи кетма-кетлик халокатли холатларга олиб келмаслиги керак.

- тил коидалари буйича конуний булмаган барча конструкциялар топилиши ва белгиланиши шарт.
- дастурчининг хатолари тугри диагностика килишиши керак, ва трансляторнинг ишини хеч кандай оркага сурмаслиги шарт.

Биринчи иккита талаб сузсиз бажарилиши шарт, охирги хусусият эса бу истакдир, чунки амалиётда хар кандай холат булиши ва баъзи хатоликлардан тулик кочишнинг иложи булмай колиши хам мумкин.

Назорат саволлари

1. Синтаксис тахлил жараёнида юз берган хатоликларни тиклашда кулланилиши мумкин булган йуналишлар хакида маълумот беринг. Уларнинг кайси бири фойдалирокдир?
2. Кандай белгилар тиклаш белгилари хисобланадилар?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com

Маъруза №14.
Мавзу: Постфикс ёзув.
Режа:
1.Дастурларни ички тасвирлаш усуллари.
2. Ассемблер коди ва машина командалари.
3. Постфикс ёзув.
4.Семантик тахлил боскичлари

Калит сузлар.

- **Инфикс ёзув**
- **Префикс ёзув**
- **Постфикс ёзув**
- **Польша ёзувлари**
- **Қавссиз ёзувлар**

1.Дастурларни ички тасвирлаш усуллари.

Синтаксис дарахт – бу структура синтаксис тахлилчини иши натижасини ифодалайди. У кирувчи тил конструкцияси синтаксисини акс эттиради ва амалларнинг тулик узароалокасини узида жамлайди.

Камчилиги: Синтаксис дарахтлар мураккаб богланган структурага эга булганликлари сабабли, улар тривиал куринишда натижавий дастурнинг чизикли кетма-кетлигига айлантирила олмайдилар.

Очик номланган натижали купманзилли код.(Тетрадалар). Тетрадалар 4 та таркибий кисмдан иборат амаллар ёзувини ифодалайдилар: <амал>(<1операнд><2операнд><натижа>). Тетрадалар чизикли командалар кетма-кетлиги булиб, бири биридан кейин хисобланадилар. Агар операндлардан кайсидир бири йук булса, лекин у ёки тушириб колдирилиши ёки буш операнд билан узгартирилиши мумкин. Хисоблаш натижаси тушириб колдирилиши мумкин эмас. Тетрадалар чизикли боглик булганликлари сабабли, улар учун тетрадалар кетма-кетлигини командалар кетма-кетлигига айлантирадиган тривиал алгоритмни ёзиш етарлидир. Бундай ифодалаш куриниши хисоблаш тизимининг архитектурасига боглик эмас.

Машинага богликмас куриниш

1)Триадага нисбатан купрок хотира талаб этади. 2) амаллар орасидаги узароалокани аник ифодаламайди.

Очикмас номланган натижали купманзилли код. (триадалар)

<амал>(<1операнд><2операнд>)

Триадалар 3 холатли амаллар ёзувларини ифодалайдилар. Триадаларнинг хусусиятлари шуки, бир ёки иккала операнд бошка триадага ссылка булиши мумкин. Тетрадалардан фаркли равишда бу ерда хотирани таркатилишига жавоб берувчи алгоритмни булиши талаб этилади (хисоблашларни оралик натижаларини сакловчи). Триадалар узини тасвирлаш учун камрок хотира

талаб этадилар, улар узлари уртасидаги амалларнинг узаро алокаларини очик ифодалайдилар.

Ютуклари:1) триадалар командаларнинг чизикли кетма-кетлигини ташкил этиб, улар учун триадаларни кетма кетлигини натижавий дастур командалари кетма-кетлигига айлатириш алгоритмини ёзиш етарлидир. Бу ерда махсус хотирани таркатишга жавобгар ва хисоблашларни оралик натижаларини сакловчи алгоритмдан фойдаланиш зарур.

Дастурларни ички тасвирлашнинг машинага богликлик куриниши. Ютуклари: узини ифодалаш учун кам хотира талаб этади, амалларни узаро алокасини аник ифодалайди.

2.Ассемблер коди ва машина командалари.

Машина командалари шу билан кулайки, фойдаланишда дастурнинг ички тасвирланиши тулик объект кодига мос келади ва мураккаб айлантиришлар талаб этилмайди.

Ассемблер командалари бу машина командаларининг ёзилиш куринишидир. Бу ерда акс эттириш ва амалларнинг алокаси учун кушимча структуралар талаб этилади. Бу форма – дастурларнинг машинага богликлик куринишидир.

Амалларнинг тескари польша ёзуви. Бу амалларнинг постфикс ёзилишидир. Амалларнинг инфекс ёзувига нисбатан, польша ёзуви операндлари уз тартиби буйича, амал ишоралари эса катъий бажарилиш тартибида келади. Хисоблаш учун стекдан фойдаланилади. Камчилиги: Биз стекнинг юкориси билангина ишлай олишимиз сабабли, ифодаларни оптималлашнинг иложи йук.Ифодалар чапдан унгга караб чикилади ва элементлар куйидаги коида буйича кайта ишланади:

Алгоритм:

1. Агар операнд белгиси учраса у холда у стека жойланади (стекнинг юкориси булиб хисобланади)
2. Агар унар амал белгиси учраса у холда битта операнд стекдан тортиб чикарилади, амал бажарилади ва натижа стекнинг юкорисига жойлашади.
3. Агар бунар амал белгиси учраса у холда иккита операнд стекдан тортиб чикарилади, амал бажарилади ва натижа стекнинг юкорисига жойлашади.

3.Постфикс ёзув.

Кундалик амалиётда учрайдиган кавсларни уз ичига олган оддий арифметик ифодаларни **инфекс ифодалар** деб аталади, чунки амал ишораси операндлар орасида жойлашади. Харакатларнинг бажарилиш тартиби бундай ифодаларда амалларнинг катталик даражаси ва кавслар билан белгиланади. Бундай ифодаларни хисоблаш ва компиляциялаш уларни аввалдан амалларни бажарилиш тартибини аниклаш максадиди тахлил килишни англатади.Арифметик ифодаларни кавсларсиз ёзиш усуллари мавжудки, уларда характлар тартиби ифодадаги амал белгилари тартиби билан берилади. Ёзувларнинг бундай куринишини **польша ёзувлари** ёки **кавссиз ёзувлар** деб аталади. Польша ёзуви **префикс** булиши мумкин, бу холда амал ишораси операндлардан аввал келади, ва **постфикс** булиши мумкин, бу холда амал ишораси операндлардан кейин келади. Бундай

кавссиз ифодаларни хисоблаш ва компиляциялаш кавсли ифодаларга караганда соддарокдир, чунки амаллар ифодаланиш тартибида бажарилади ва аввалдан тахлил талаб этилмайди.

Префикс польша ёзуви (ПрПЗ) куйидагича аникланади.

Агар инфикс ифода Е битта а операнддан иборат булса, у холда ПрПЗ ифода Е бу а .

Агар инфикс ифода Е1*Е2, бу ерда * амал белгиси, Е1 ва Е2 операндлар учун инфикс ифода, у холда ушбу ПрПз *Е1'Е2' , бу ерда Е1',Е2' – Е1 ва Е2 ифодаларнинг ПрПЗ идир.

Агар (Е) инфикс ифода булса, ушбу ифоданинг ПрПЗи ПрПз Е дир.

Санаб утилган коидалар берилган инфикс ифоданинг ПрПЗни куриш тартибини аниклайдилар. Масалан, (а+б)*(с-д) ифода учун ПрПЗ ни куйидагича куриш мумкин.

Биринчи бажарилувчи амалнинг операндларини аниклаймиз.

Е1= (а+б) ва Е2 = (с-д).

Аниклашларга кура префикс ёзув Е1*Е2 бу *Е1'Е2' бу ерда Е1',Е2' – Е1 ва Е2 ифодаларнинг ПрПЗ идир. Бу ифодалар учун префикс ёзувларни куришни бажарамиз.

Е1' = +аб, Е2'= -сд ва натижада куидаги куринишни оламиз:

*+аб-сд

Постфикс ёзувда эса амал ишоралари операндлардан кейин куйилиши билан фаркланади. Масалан, инфикс ёзув (А+В) га постфикс ёзув АВ+ мос келади.

Постфикс польша ёзуви (ПоПЗ) куйидагича аникланади.

Агар инфикс ёзув Е битта а операнддан иборат булса, у холда Е ни ПоПЗ а булади.

Агар инфикс ифода Е1*Е2, бу ерда * амал белгиси, Е1 ва Е2 операндлар учун инфикс ифода, у холда ушбу ПоПз Е1'Е2'* , бу ерда Е1',Е2' – Е1 ва Е2 ифодаларнинг ПоПЗ идир.

Агар (Е) инфикс ифода булса, ушбу ифоданинг ПоПЗи ПоПз Е дир.

Худди юкоридагидек мисолга ПоПЗни курамиз.

(а+б)*(с-д)

Ташки амалларнинг операндларини белгилаймиз.

Е1= (а+б) ва Е2 = (с-д).

Буларнинг постфикс ёзувларини топамиз:

Е1' = аб+, Е2'= сд-

Ифодага куйиб Е1'Е2'* ва натижада аб+сд-* га эга буламиз.

Ифодаларнинг постфикс ёзуви иккита кучли хусусиятга эга булиб, уларни шунинг учун хам компиляторларда кенг кулланилади.

Ихтиёрий ифодани ёзиш учун кавслар шарт эмас, чунки амалда иштирок этувчи оператор операндан кейин келади, ва операндларни курсатишдаги ноаниклик булмайди. Масалан, (А+В)+С постфикс ёзувда АВ+С+, А+(В+С) эса постфикс ёзувда АВС+ куринишда ёзилади.

Навбатдаги операторни укиш вактида мос операндлар укиб булинган буладилар ва оператор бошка кушимча бнрилганларни укишсиз бажарилиши мумкин. Юкорида айтилганлар бинар амалларга хам тегишли булиб уларни унар амалларга хам куллаш кийин эмас. Унар оператор (тескари ~) ни аргументдан кейин куйиш етарлидир. Мсаслан инфикс ёзув ~A A~ куринишда булади, ~(A+B) ифода эса AB+~ куринишга эга булади.

Юкоридаги хусусиятларга асосан постфикс куринишдаги ифода куйидаги оддий алгоритм оркали хисобланиши мумкин.

```
while(lex!=NULL)        //  ифодада лексемалар колмагунича
{
  lex= getNextLex( );     // кейинги лексемани укиш
  if(isOperand(lex))       // лексема, агар операнд булса
    push(lex);               //  лексемани стекга ёзиш
   if(isOperator(lex))    // лексема, агар оператор булса
     push( performOperation(lex, pop( ), pop( ) ));
/* охирги стекка ёзилган лексемалар курсатаётган амалларни бажариш ва  ушбу элементларни амал натижаси билан алмаштириш;
*/
}
```

Хисоблашлар натижасида ифоданинг киймати стекнинг ягона элементи булиб колади. Бундай стек билан амаллар бажарувчи алгоритм ихтиёрий компиляторнинг асосини ташкил этади. Купинча синтаксис тахлил блоки дастурни постфикс куринишга айлантиради, ундан кейин код генератори объект дастурни юкоридаги усул билан ифодаларни караб чиккан холда куради. Куриниб турибдики, постфикс ёзувларни хисоблаш кийинчилик тугдирмайди, лекин инфикс ёзувларни постфиксга айлантириш мураккаброк иш. Фараз килайлик тил махсулоти куйидаги куринишга эга булсин.

A B и C

Бу ерда A,B ва C нотерминал белгилардир, и эса терминал белгидир. Амал ишоралари терминал белгилар деб хисоблаймиз. У холда бу махсулот нотерминал A инфикс ёзув эканлигини ва унда B ва C «и» оператор билан биргаликда катнашаётганлигини билдиради. Худди шундай ифоданинг постфикс куриниши хам B ва C операндлардан ташкил этилади ва улардан сунг «и» оператори келади. Шундай килтб бундай махсулот учун постфикс ёзув куйидаги куринишга эга булади.

B учун постфикс ёзув

C учун постфикс ёзув

и

Агар барча терминал белгилар чикувчи файлга шу тартибда чикарилсалар, у холда барча тилнинг фразалари постфикс куринишида ифодаланадилар.

4.**Семантик тахлил боскичлари**

1. Кирувчи дастурдаги кирувчи тилнинг семантикасини текшириш. Ушбу текширув дастурнинг кирувчи занжирларини кирувчи дастурлаш тили семантикасини талабларига мос куйишдан иборат.
а:=в+с
2. Компиляторда дастурнинг ички тасвирланишини кирувчи тилнинг семантикаси ёрдамида ноаник курсатилган операторларда кушиш. Ушбу харакатлар ифодаларнинг операндлари турларини айлантириш ва функциялар, процедураларни параметрларини узатиш билан боглик.
3. Дастурлаш тилларининг кирувчи тил билан тугридан тугри богланмаган элементар семантик нормаларини текшириш.

Ушбу текширув купгина замонавий дастурлаш тиллари курсатадиган сервис функция хисобланади. Ушбуни бажарилиши мажбурий шарт эмас.
4. Тилни лексик улчовларини идентификацияси.

Узгарувчиларни, тоифаларни, процедураларни, функцияларни ва бошка дастурлашнинг лексик тилларни идентификацияси– бу берилган объектлар ва уларнинг кирувчи дастур матнидаги исмлари уртасидаги бир кийматлиликни урнатишдир.

Назорат саволлари.

1. Ифодаларнинг инфикс, постфикс ва префикс ёзувларини фаркларини айтиб беринг.
2. Компиляторларда фойдаланилиши нуктаи назаридан постфикс ёзувнинг устунлиги нимада?
3. Инфикс ёзувларни постфикс ёзувларга айлантиришнинг формал усуллари мавжудми?
4. Постфикс ёзувида ифодаланган куйидаги ифодани а=2; в=4; кийматларда хисобланг (барча узгарувчилар ва сонли константалар 1 узунликка эга).

   1а+1а3--*в+
5. Ифоданинг инфикс ёзувини постфикс ёзувига айлантиринг.

   (а+(а-в))*(в+2-а)

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.

5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru

Маъруза №15.

Мавзу:Семантик тахлил ва кодни генерациялаш усуллари.

Режа:

1.Кодни оптималлашнинг умумий тамойиллари.

2. Кодни генерациялаш усуллари.

Калит сузлар.

- **Семантик тахлил**
- **Компилятор**
- **Кирувчи занжир**
- **Операнд**
- **Функциялар**
- **Процедура**
- **Параметр**

1.Кодни оптималлашнинг умумий тамойиллари.

Оптимизация бу фойдалирок натижали объект дастур олиш максадида компьютер дастуридаги амалларнинг узгартириш ва тартибга солиш билан боглик кайта ишлашдир. Оптимизация бир неча марта бажарилиши мумкин, код генерациясини тайёрлаш фазаси буйича ва кодни генерациялаш фазаси буйича.Натижавий дастурннинг фойдалилик курсаткичи булиб куйидаги критерийлардан фойдаланилади: 1)натижавий дастурнинг бажарилиши учун зарур булган хотира хажми 2) дастурнинг бажарилиш тезлиги.

Айлантиришларни оптималлашни икки асосий куринишини фарклайдилар: 1) кирувчи дастур матнини натижавий объект кодига боглик булмаган холда унинг ички тасвирланишини куринишида айлантириш. 2) берилган айлантиришлар максадли хисоблаш тизимининг архитектурасидан боглик эмас. Улар аввалдан яхши таниш булган математик ва мантикий айлантиришларга асосланган. 3) натижавий объект дастурни айлантириш.

Ушбу гурух айлантиришлари максадли хисоблаш тизимининг архитектурасидан боглик.

Оптималлаш куйидаги синтаксис конструкциялар учун бажарилиши мумкин: 1)дастурнинг чизикли булаклари; 2) мантикий ифодалар 3)цикллар 4)процедура фукцияларини чакириклари

2.Кодни генерациялаш усуллари.

Кодни ички ёзувларининг бир хил фрагментлари (постфикс ёзувлари амаллари, туртлик ва бошкалар) машина тилининг бир хил буйрукларини ифодалайди. Масалан, код генерацияланаётган PLUS_OP туртлик, агар процессорда барча амаллар регистр-аккумлятор устида бажарилса, хар доим куйидаги кодни генерациялайди:

LOAD регистр , операнд 1

ADD регистр, операнд 2

STORE регистр, натижа

Машина командаларининг бу кетма-кетлиги коррект, лекин оптимал эмас. Масалан, куйидаги гап

X:=X+Y*Z олтита команда оркали амалга оширилади:

LOAD регистр, Y (туртлик (MULT_OP,Sy,Sz,T1))

MUL регистр, Z

STORE регистр, T1

LOAD регистр, X (туртлик (ADD_OP,Sx,T1,Sx))

ADD регистр, T1

STORE регистр, X

Худди шунингдек, ушбу натижага келтирувчи куидаги дастурни куриш мумкин.

LOAD регистр, Y

MUL регистр, Z

ADD регистр, X

STORE регистр, X

Ушбу усул билан генерацияланаётган код хар доим тугри хисобланади, лекин хар доим хам оптимал эмас. Шунинг учун кодни хисоблашларни аниклигига таъсир курсатмай туриб, узгартириш имконини берувчи курилмаларга эга булиш керак.

Хар бир туртликка купгина холларда ягона машина командалари кетма-кетлиги мос келади, код генератори купинча хар бир туртлик буйича кисмдастурлар туплами мос келади.

Назорат саволлари.

1. Семантик тахлил босқичининг вазифаси нмалардан иборат?
2. Семантик тахлил босқичлари хақида маълумот беринг.
3. Кодни оптималлашнинг қандай усулларини биласиз?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. А.Левин. Самоучитель полезных программ. Питер. Санкт-Петербург, 2002.
5. Карпов Б.И. Delphi: Специальный справочник. – СПб: Питер, 2001-648с.
6. Карпов Б.И. Visual Basic Специальный справочник. – СПб: Питер, 2000-415с.
7. Карпов С.Ю. Теория автоматов. Учебные пособия для вузов. –СПб: Питер, 2003.-201с.
8. WWW.codecrojekt.ru
9. WWW. master.ru
10. WWW.bdn_borland.com
11. http://microsofft.com

Маъруза №16.

Мавзе: Ички формалар.

Режа:

1. Тўртлик формаси.

2.Кодни генерациялашда «тўртлик»ни қўллаш

Калит сузлар.

- **Постфикс ёзув**
- **Дастур коди**
- **Генерация**
- **Тўртлик**

1. Тўртлик формаси.

Постфикс ёзувдан дастур кодини куриш мумкин, лекин бундай ёзув формасини оптималлаштириш мураккаб иш. Купгина компильяторлар дасурнинг объект кодини куриш учун оптималлаш учун кулай булган ички формалардан фойдаланадилар. Генерация килинаётган коднинг энг куп таркалган ички тасвирлашни формаларидан бири бу **туртликдир.**

Туртлик –бу туртта элементдан ташкил топган объектдир: амаллар, иккита операнд ва натижалар. Агар амал бажариш натижасида кандайдир узгарувчини киймати хисобланса, у холда бундай туртликни куриш унчалик мураккаб эмас. Масалан: X:= Y+Z гап куйидаги туртлик оркали ифодаланади.

(PLUS_OP, Sy, Sz,Sx), бу ифода Sy белгилар жадвали билан ячейкада аникланган узгарувчини Sz ячейкада аникланган узгарувчи билан (PLUS_OP) кушиб ва натижани Sx ячейкада саклашни англатади. Энди бошлангич гап мустакил бирлик сифатида ифодаланади, уни код генератори жойлашиш манзилидан катьи назар кайта ишлай олади. Шундай килиб,

оптимизатор амаллар кетма-кетлигини кодни генерациялаш жараёнини мураккаблаштирмасдан узгартириши мумкин.

2.Кодни генерациялашда «тўртлик»ни қўллаш

Унар операторлар учун туртликнинг иккинчи операндини майдонини игнорироват килиш мумкин, иккитадан ортик операндларни талаб киладиган амалларни эса бир неча туртликлардан ташкил топган кетма-кетликлар куринишида ифодалаш мумкин.

Масалан, куйидаги операторни X:= F(A,B,C,D) учта туртлик куринишидаги гурух сифатида ёзиш мумкин.

(F1,A,B, T1)
(F2, T1,C, T2)
(F, T2,D,X)

F1 ва F2 функциялар оралик хисоблашларни амалга оширадилар, T1 ва T2 ячейкалар эса ушбу харакатларнинг натижаларини саклаш учун мулжалланган. Дастурни фактик куриш вактида код генератори объект кодида F1 (бу амал учун F2 ва F амаллар оркали) амални тугри ифодалаши мумкин.

Яна оралик ячейкалардан фойдаланишга боглик мисол караймиз.

Фараз килайлик куйидаги постфикс ёзувли гап берилган булсин.

SxSxSySz*+:=

Бу гапга Y ва Z купайтириш амаллари, натижани X билан кушувчи ва X узгарувчига олинган суммани узлаштириувчи амаллар киради. Туртликни генерациялаш вактида купайтиришни амалга оширувчи учун ушбу оралик натижани сакловчи ячейка керак булади. Бу холатда, вактинчалик ва ички узгарувчи ташкил этилади деб фараз киламиз. Шундай килиб, каралаётган гап куйидаги кетма-кетликда ифодаланади.

(MULT_OP, Sy, Sz,T1)

(ADD_OP,Sx,T1,Sx),

Бу ерда биринчи туртлик Y ни Z га купайтириш ва натижани T1 ячейкага ёзишни, иккинчи туртлик эса X узгарувчини Y*Z амал натижасини сакловчи T1 узгарувчи билан кушишни ва суммани X га ёзувни аниклайди.

Назорат саволлари

1. Генерация килинаётган колни ички ифодаланишида «туртлик» кандай булаклардан ташкил топади?
2. Кодни генерациялашда «туртлик»ни куллашнинг ютуклари нималарда куринади?

Фойдаланилган адабиётлар

1. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. –СПб: Питер, 2003.-396 с.
2. Афанасьев А.Н. Формальные языки и грамматики: Учебная школа: УлГТУ, 1997. – 84 с
3. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции -: Мир, 1979.-487с.
4. Компаниец Р.И. Системное программирование. Основы построения трансляторов. СПб.:Корна принт., 2000. -256 стр.
5. Дьяконов В.Ю. Системное программирование. Высш.шк.. 1990. -221 с.
6. WWW.codecrojekt.ru
7. WWW. master.ru
8. WWW.bdn_borland.com
9. http://microsofft.com