ЎЗБЕКИСТОН АЛОҚА, АХБОРОТЛАШТИРИШ ВА ТЕЛЕКОММУНИКАЦИЯ ТЕХНОЛОГИЯЛАРИ ДАВЛАТ ҚЎМИТАСИ
ТОШКЕНТ АХБОРОТ ТЕХНОЛОГИЯЛАРИ УНИВЕРСИТЕТИ

«Дастурий инжиниринг» факультети
«АТДТ» кафедраси
«ОЙДТ» фанидан

РЕФЕРАТ

Мавзу: Синфларда ворислик Инкосульатция

Бажарди: 216-11 гуруҳ талабаси
Турсунова Ш.
Қабул қилди: Раҳмонова М

Тошкент 2013-2014

Кириш

Сўнги йилларда компьютернинг дастурий таъминоти ривожланиши асосий йўналишларидан бири бу объектга йўналтирилган дастурлаш соҳаси бўлди. Обектга йўналтирилган операцион тизимлар (Масалан, Windows), амалий дастурлар ва объектга йўналтирилган дастурлаш (ОЙД) тизимлари ҳам аммвийлашди.

Биринчи ОЙД элементи Симула-67 (1967 й., Норвегия) тили бўлди. Turbo PASCAL да 5,5 версиясидан бошлаб ОЙД воситалари пайдо бўлди. Turbo PASCAL нинг ривожи якуни якуни сифатида BORLAND филмаси томонидан DELPHI дастурлаш тизими яратилиши бўлди. Ушбу система ёрдамида тез ва осон мураккаб бўлган график интерфейсни дастурлаш имконияти мавжуд. 1991 йилда Visual BASIC нинг 1 версиясидан бошлаб бу тил тўлалигича объектга йўналтирилди (1997 йил).

ОЙД тилларидан яна бири 1995 йилда Жеймс Гослинг бошчилигида Sun Microsystems компаниясида яратилган JAVA тилидир. Уни ишлаб чиқишда махсус ўрганиш талаб қилмайдиган, содда тилни мақсад қилинган.

JAVA тили максимал даражада C++ тилига ўхшаш бўлиши учун яратилган яратилган JAVA Интернет учун дастурлар тайёрлашнинг идеал воситасидир. Сўнги йилларда Microsoft компанияси томонидан C++давомчиси сифатида C# (Си шарп) тили яратилди.

1985 йилда Bell Labs (АҚШ) лаюораторияси C++ дастурлаш тили яратилганлигини хабарини берди. Бугунги кунда бу тил ОЙД тиллари орасида машҳурдир. Бу тил ёрдамида исталган машина учун – шахсийдан то суперкомпьютерларгача дастурлар ёзиш мумкин. Бу тилнинг асосчиси Бьорн Страуструпдир.

Масалани қўйилиши

ЁТОҚХОНА (номи, ХОНАЛАР, СТУДЕНТЛАР, ФАКУЛЬТЕТЛАР)Хонага тегишли ҳамма маълумотлар.синфи берилган.Фойдаланувчи структура массивини яратинг. Оқимда(файлда) уларни сақлаш функциясини яратинг. Дастурдан фойдаланинг. Кўрсатилган кўрсатилган структура массивини файлдан ўқиш ва чиқариш функцияларини яратинг Берилган критерий асосида фапйлдаги маълумотларни корректировка (алмашиш) қилувчи функция яратинг. Дастурни яратинг ва мавжуд бўлган файлни кўриб чиқинг.

Ишдан мақсад: Стандарт библиотекадаги оқимли синфлар объектларидан фойдаланган холда киритиш ва чиқариш дастурлашни ўрганиш.

Назарий қисм

Синфлар ва объектлар

Синтаксис бўйича, C++ ва JAVA да синф – бу мавжуд бўлган типлар асосида янги яратилган структурланган тип.

Синф таърифи содда шакли:

<синф_типи> <синф_номи>{<синф_компонентлари_рўйхати>};

бу ерда:

синф_типи –**class, struct, union** хизматчи сўзларидан бири; Java да фақат **class** ишлатилади.

синф_номи – идентификатор;

синф_компонентлари_рўйхати – синфга тегишли маълумотлар ва функциялар таърифи.

Функция – бу объектлар устида бажариладиган операцияларни аниқловчи синф усули.

Маълумотлар – бу объект структурасини хосил қилувчи майдон.

Усуллар синфдан ташқарида аниқланганда уларнинг номларини квалификация қилиш (ихтисослаштириш) керак. Усулнинг кўримлилик сохасини аниқлайдиган унинг бундай квалификация синтаксиси қуйидаги кўринишга эга:

<синф номи>::<усул номи>

Синф ичида аниқланган усуллар кўзда тутилган бўйича жойлаштирилувчи (**inline**) функция хисобланади. Синф ташқарисида аниқланган усулларни ошкор равишда жойлаштирилувчи деб таърифланиши лозим.

Синф объекти (синф нусхаси) ни таърифлаш учун қуйидаги конструкциядан фойдаланилади:

<синф_номи> <объект_номи>;

Объект орқали майдонларга ва усулларга қуйидагича мурожжат қилиш мумкин:

< объект_номи >. <майдон_номи>

< объект_номи >. <усул_номи>

Компоненталарга мурожат хуқуқлари

Компоненталарга мурожаат хукуки мурожаат спецификаторлари ёрдамида бошкарилади: **public, private, protected**.

Умумий (public) компоненталар дастурни ихтиёрий қисмида мурожаат хуқуқига эга. Улардан, ихтиёрий функция ушбу синф ичида ва синф ташқарида фойдаланса хам бўлади.

Хусусий (private) компоненталар синф ичида мурожаат хуқуқига эга, лекин синф ташқарисидан эса мурожаат килиш мумкин эмас. Компоненталардан ушбу улар тавсифланган синфдаги функция - аъзолари ёки “дўстона”- функциялар орқали фойдаланиш мумкин.

Химояланган (protected) компоненталар синф ичида ва хосила синфларда мурожаат хуқуқига эга.

Агар синф таърифида **class** сўзи ишлатилган бўлса хамма компоненталари хусусий хисобланади, агар **struct** всўзи ишлатилган бўлса хамма компоненталар умумий хисобланади.

Конструктор

Конструктор - бу синф объектларини автоматик инициализация килиш учун ишлатиладиган махсус компонентали функция.

Конструкторлар кўриниши қуйидагича булиши мумкин:

<Синф номи> (<формал параметрлар руйхати>)

{<конструктор танаси>}

Бу компонента функция номи синф номи билан бир хил булиши лозим.

Дастурчи томонидан кўрсатилмаган холда хам new оператор ёрдамида синф объекти яратилганда ёки хотирада жойлашганда конструктор автоматик равишда чақирилади.
Конструктор объект учун хотирада жой ажратади ва маълумотлар – синф аъзоларини инициализациялайди.
Конструктор бир нечта хусусиятларга эга:

- Конструкторлар учун кайтарилувчи типлар, хатто void типи хам кўрсатилмайди
- Конструктор адресини хисоблаш мумкин эмас. Конструктор параметри сифатида уз синфининг номини ишлатиш мумкин эмас, лекин бу номга кўрсаткичдан фойдаланиш мумкин.
- Конструкторлар ворисликга эга эмас.

Конструкторлар ихтиёрий синфлар учун доимо мавжуд, лекин агарда у кўрсатилган холда тавсифланмаган бўлса, у автоматик равишда яратилади. Кўрсатилмаган холда параметрсиз конструктор ва нусха конструктори яратилади. Агарда конструктор очиқ холда тавсифланган бўлса, унда кўрсатилмаган холда конструктор яратилмайди. Кўрсатилмаган холда умумий (public) конструкторлар яратилади.

Объектлар массиви

Объектлар массиви таърифлаш учун синф кўзда тутилган (параметрсиз) конструкторга эга бўлиши керак.
Объектлар массиви кўзда тутилган конструктор томонидан, ёки хар бир элемент учун конструктор чақириш йўли билан инициализация қилиниши мумкин.

```
class complex a[20]; //кўзда тутилган параметрсиз конструкторни чақириш
class complex b[2]={complex (10),complex (100)};//ошкор чақириш
```

finalize

Java тилида **finalize** номли усуллар киритиш имконияти мавжуд. Бу номли усуллар C++ тилидаги (калит белги ~) ва Delphi тилидаги (калит сўз **destructor**) деструкторларга мосдир. Java бажариш мухити хар гал объектни ўчиришда шу услни чақиради.

This кўрсаткичи

Агарда конкрет объектга ишлов бериш учун синф аъзоси – функция чақирилса, унда шу функцияга объектга белгиланган кўрсаткич автоматик ва кўрсатилмаган холда узатилади. Бу кўрсатқич **this** исмига.

Синфни қуйидаги тасвирлаш мумкин:

```
class Point { int x, y;
Point(int x, int y) {
this.x = x;
this.y = y;
} }
```

Синфнинг статик компоненталари

Синф компанентаси ягона булиб ва хамма яратилган объектлар учун умумий булиши учун уни статик элемент сифатида таърифлаш яъни **static** атрибути оркали таърифлаш лозимдир. Объектларни яратишда синф статик маълумотлари такрорланмайди, яъни хар бир статик компонентлар бирдан-бир кўринишда мавжуд.

Статик усуллар фақат бошқа статик усулларга тўғридан тўғри мурожаат қилиши мумкин ва уларда this иловасидан фойдаланиш мумкин эмас. Ўзгарувчилар хам static типга эга бўлиши мумкин, бу холда уларга глобал ўзгарувчиларга ўхшаб дастур ихтиёрий қисмидан мурожаат қилиш мумкин. Статик усуллар ичида ностатик ўзгарувчиларга мурожаат қилиш мумкин эмас.

Синф статик маълумотларга фақатгина объект исми орқали мурожаат этиш мумкин. **<объект_номи>.<компонента_номи>**

Масалан:

```
complex a; a.count=5;
```

Лекин, статик компонетларга синф объекти аниқланмаган холда хам мурожаат этиш мумкин. Статистик компонентларга нафақат объект исми, балки синф исми орқали хам мурожаат этиш мумкин.

<синф_номи>. <компонента_номи>

Лекин шундай мурожаат фақатгина *public* компонентларга тегишли.

private статик компонентларга ташқаридан мурожаат этишда **функция – статик компонентлардан** фойдаланилади.

Бу функцияларни синф исми орқали чақириш мумкин.

<синф_номи> .: <статик_функция_номи>

Синфлар ва объектлар билан ишлаш хусусиятлари

Runtime

Runtime синфи Java интерпретаторини инкапсуляция қилади. Бу синф объектини яратиш мумкин эмас, лекин статик усулидан фойдаланиб ишлаб турган объектга мурожжаат қилиш мумкин. Одатда апплетлар ва бошқа дастурлар синф усулини чақирганда SecurityException истисноси хосил бўлади. Runtime объектини тўхтатиш учун exit(int code) усулини чақириш етарли.

Хотирани бошқариш

Java тилида хотира автоматик тозаланса хам дастур эффективлигини текшириш учун "уюм" хажмини аниқлаш ва эркин хотира хажмини хисоблаш мумкин. Бу маълумотни аниқлаш учун totalMemory ва freeMemory усулларидан фойдаланиш мумкин.

Зарур бўлганда хотира тозаловчини gc усулини чақириб автоматик ишга тушириш мумкин. Дастурга зарур хотирани хисоблаш учун аввал gc, сўнгра free-Memory усулларини чақириш керак. Шундан сўнг дастурни ишга тушириб freeMemory усули чақирилса, дастур қанча хотира ишлатишини аниқлаш мумкин.

Бошқа дастурларни бажариш

Хавфсиз мухитларда Java тилидан бошқа жараёнларни ишга тушириш учун фойдаланиш мумкин. Бунинг учун exec усулининг бир неча шаклларидан фойдаланиш мумкин. Усулга дастур номи ва бир неча параметрлар узатилади.

System

System синфида турли глобал функциялар ва ўзгарувчилар мавжуд. Масалан System.out.println() усули.

Бундан ташқари **currentTimeMillis** усули тизимли вақтни 1970 йил 1 январидан бўлиб ўтган миллисекундларда қайтаради.

Массивдан нусха олиш учун **arraycopy** усулидан фойдаланилади. Қуйида бир массивдан иккинчисига нусха олишга мисол келтирилган.

```
class ACDemo {
static byte a[] = { 65, 66, 67, 68, 69, 70, 71, 72, 73, 74 };
static byte b[] = { 77, 77, 77, 77, 77, 77, 77, 77, 77, 77 };
public static void main(
String args[]) {
System.out.println("a = " + new String(a, 0));
```

```
System.out.println("b = " + new String(b, 0));
System.arraycopy(a, 0, b, 0, a.length);
System.out.println("a = " + new String(a, 0));
System.out.println("b = " + new String(b, 0));
System.arraycopy(a, 0, a, 1, a.length - 1);
System.arraycopy(b, 1, b, 0, b.length - 1);
System.out.println("a = " + new String(a, 0));
System.out.println("b = " + new String(b, 0));} }
```

Мухит хоссалари

Java бажариш мухити Properties синфи объекти орқали мухит ўзгарувчиларига мурожаат қилишга имкон беради. Хоссалар тўлиқ рўйхатини олиш учун System.getProperties() усулини чақириш лозим.

Жадвал

Система стандарт хоссалари

Исм	Қиймат	Апплет учун рухсат
Java.version	Java интерпретатора версияси	ха
Java.vendor	Фойдаланувчи киритган идентификатор қатори	ха
java.vendor.url	Ишдаб чиқарувчи URLи	ха
java.class.version	Java API версияси	ха
java.class.path	CLASSPATH ўзгарувчиси қиймати	йўқ
java.home	Java мухити инсталляция қилинган каталог	йўқ
java.compiler	Компилятор JIT	йўқ
os.name	Операцион тизим номи	ха
os.arch	Дастур бажарилаётган компьютер архитектураси	ха
os.version	Web-тугун операцион тизими версияси	ха
file.separator	Платформага боғлиқ файл ажратувчилари (/ ёки \)	ха
path.separator	Платформага боғлиқ йўл ажратувчилари (: ёки ;)	ха
line.separator	Платформага боғлиқ сатр ажратувчилари (\n ёки \r\n)	ха

user.name	Жорий фойдаланувчи исми	йўқ
user.home	Фойдаланувчи каталоги	йўқ
user.dir	Жорий ишчи каталог	йўқ
user.language	2-символли махаллий тил коди	йўқ
user.region	2-символли мамлакат тил коди	йўқ
user.timezone	Кўзда тутилган вақт зонаси	йўқ
user.encoding	Кўзда тутилган бўйича белгилар коди	йўқ
user.encoding.pkg	Махаллий символларни Unicode кодига ўтказиш учун конверторлар пакети	йўқ

Date

Сана ва вақт билан ишлаш учун Date синфидан фойдаланилади. У орқали сана, йил , ой, хафта кунига, соат, минут, секундга мурожаат қилиш мумкин. Бу синф турли конструкторлари мавжуд. Энг соддаси — Date() —объектни жорий сана ва вақт билан инциализация қилади. Қолган уч конструктор қўшимча имкониятларга эга.

- Date(year, month, date) — кўрсатилган санани ўрнатади, вақт 00:00:00 (тунги) қийматга эга бўлади.
- Date(year, month, date, hours, minutes) — кўрсатилган сана ва вақтни ўрнатади, секунд 0 қийматни олади.
- Date(year, month, date, hours, minutes, seconds) — энг тўлиқ кўриниши, сана ва вақт хамда секундлар хам ўрнатилади.

get ва set

Класс Date синфи объект атрибутларини ўрнатиш учун усулларга эга. Бу оилага кирувчи усулар — getYear, getMonth, getDate, getDay, getHours, getMinutes и getSeconds — бутун қиймат қайтаради.

Date синфи қийматини getTime усули long типидаги сон сифатида қайтаради. Бу сон 1970 йил 1 январидан ўтган миллисекундларга тенг.

Солиштириш

Date типидаги икки объектни солиштириш учун, санани миллисекундларга айлантириш лозим. Date синфи тўғридан тўғри солиштириш учун уч усулга эга: — before, after ва equals. Масалан

new Date(96, 2, 18).before(new Date(96, 2, 12)

true қайтаради, чунки ойнинг 12-куни 18-кунидан олдин келади.

Сатр ва сана

Объекты Date объектини турли форматдаги матнга кон иможно конвертация қилиш мумкин. Аввалambор toString усули Date объектини қуйидагича сатрга алмаштиради “Thu Feb 15 22:42:04 1996”. Кейинги toLocaleString усули санани қисқароқ сатрга алмаштиради, масалан: “02/15/96 22:42:04”. Ва нихоят toGMTString усули санани Гринвич бўйича ўртача вақт форматига ўтказади: “16 Feb 1996 06:42:04 GMT”.

Math

Math синфи геометрия ва тригонометрияда ишлатиладиган сузувчи вергулли функцияларга эга. Бундан ташқари хисоблашларда ишлатиладиган икки константа мавжуд: — E (тахминан 2.72) ва PI (тахмина 3.14159).

Тригонометрик функциялар.

Қуйида келтирилган уч функция радианларда бурчакни ифодаловчи double типидаги параметрга эга бўлиб, мос тригонометрик функция параметрини қайтаради.

- sin(double a) радианда берилган a бурчак синусини қайтаради.
- cos(double a) радианда берилган a бурчак косинусини қайтаради.
- tan(double a) радианда берилган a бурчак тангенсини қайтаради.

Кейинги тўрт функция узатилган параметр қийматига мос бурчакни радианларда қайтаради.

- asin(double r) синуси r га тенг бурчакни қайтаради.
- acos(double r) косинуси r га тенг бурчакни қайтаради.
- atan(double r) тангенси r га тенг бурчакни қайтаради.
- atan2(double a, double b) тангенси a/b га тенг бурчакни қайтаради.

Даражага кўтариш, экспонента ва логарифм функциялари

- pow(double y, double x) x даражага кўтарилган y қайтаради. Масалан, pow(2.0, 3.0) тенг 8.0.
- exp(double x) e даражаси x қайтаради.
- log(double x) x натурал лограифмини қайтаради.
- sqrt(double x) x квадрат илдизини қайтаради.

Яхлитлаш

- ceil(double a) қиймати a дан катта ёки a га тенг бўлган энг кичик бутун сон қайтаради.
- floor(double a) қиймати a дан кичик ёки a га тенг бўлган энг катта бутун сон қайтаради.
- rint(double a) каср қисми олиб ташланган double типида a қийматини қайтаради.
- round(float a) энг яқин бутун сонга яхлитланган a қийматини қайтаради.
- round(double a) энг яқин узун бутун сонга яхлитланган a қийматини қайтаради.

Бундан ташқари Math синфида int, long, float ва double типлари билан ишловчи модул олиш, минимал ва максимал қийматни топиш усуллари полиморф версиялари мавжуд:

- abs(a) a модули (абсолют қиймати) ни қайтаради.
- max(a, b) ўз аргументлари энг каттасини қайтаради.
- min(a, b) ўз аргументлари энг кичигини қайтаради.

Random

Random — псевдотасодифий сонлар генератори бўлиб, унда ишлатилган алгоритм Дональд Кнут “Дастурлаш санъати ” китобининг 3.2.1 бўлимида келтирилган. Одатда бошланғич қиймат сифатида жорий вақт олинади, бу эса қайтарилувчи тасодифий сонлар олиниши эхтимоллигини камайтиради.

Random синфи объектидан 5 турдаги тасодифий сонларни олиш мумкин. Бу тип диапазони бўйича бир текисда тақсимланган бутун сонни олиш учун nextInt усулидан фойдаланилади. Шунга ўхшаш nextLong усули long типидаги тасодифий сонни қайтаради. Бундан ташқари nextFloat ва nextDouble усуллари мос равишда float ва double типидаги, 0.0..1.0 интервалда текис тақсимланган сонларни қайтаради.

СИНФЛАРДА ВОРИСЛИК

Ворисликда мурожаат хуқуқларини бошқариш

Ворислик ўзининг барча аждодларининг хусусиятлари, маълумотлари, методлари ва воқеаларини мерос қилиб оладиган хосила синфини эълон қилиш имкониятини беради, шунингдек янги тавсифларни эълон қилиши хамда мерос сифатида олинаётган айрим функцияларни ортиқча юклаши мумкин. Базавий синфнинг

кўрсатиб ўтилган тавсифларини мерос қилиб олиб, янги туғилган синфни ушбу тавсифларни кенгайтириш, торайтириш, ўзгартириш, йўқ қилиш ёки ўзгаришсиз қолдиришга мажбурлаш мумкин.

В JAVA тилида тўғридан тўғри аждод суперсинф деб аталади.

Хосила синфни эълон қилишнинг умумлашган синтаксиси:

class <синф номи>: [<кириш хуқуқини берувчи сецификатор >] <аждод синф номи> {…}

Конструктор ва деструкторларда ворислик

Конструкторлар мерос бўлмагани учун, хосила синфни яратишда ундан мерос бўлган маълумот – аъзолари асосий(базавий) синф конструктори орқали инициализацияланиши лозим. Асосий синф конструктори автоматик равишда чақирилади ва хосила синфни конструкторидан олдин бажарилади. Асосий (базавий) синфни конструкторининг параметрлари хосила синфни конструкторни аниқлашда кўрсатилади. Шундай қилиб аргументларни хосила синфни конструкторидан асосий (базавий) синфни конструкторига узатиш вазифаси бажарилади.

Асос синф конструктори параметри хосила синф конструктори таърифида **super** калит сўзи ёрдамида кўрсатилади.

Синф объектлари пасдан тепага қараб конструкторланади: аввало асосий(базавий), кейин эса копмонент – объектлар (агарда улар мавжуд бўлса), ундан кейин эса хосила синфнинг ўзи. Шундай қилиб, хосила синфнинг объекти қуйи объект сифатида асосий (базавий) синф объектини ўз ичига олади.

Объектлар тескари тартибда ўчирилади: аввало хосила, кейин унинг компонент – объектлари, ундан кейин эса асосий(базавий) объект.

Шундай қилиб объектни ўчириш тартиби унинг конструкторлаш тартибига нисбатан тескари бўлади.

Виртуал функциялар

Виртуал функциялар механизмига бирор компонент функциянинг хар бир хосилавий синфда алохида варианти мавжуд бщлиш лозим бщлганда мурожаат қилинади. Бундай функцияларга эга синфлар **полиморф** синфлар деб аталади ва объектли дастурлашда ахлохида ўринга эга.

Виртуал функциялар кечки ёки динамик боғланиш механизми асослангандир. Асос синф хар қандай ностатик компонент

функцияси **virtual** калит сўзи ёрдамида виртуал деб эълон қилиниши мумкин.

Кечки боғланишда эрта боғланишга ўхшаб адреслар статик равишда компиляция жарёнида эмас, балким динамик дастур бажарилиши жараёнида аниқланади. Боғлаш жараёни виртуал функцияларни адреслар билан алмаштиришдан иборат. Виртуал функциялар адреслар хақида маълумот сақланувчи жадвалдан фойдаланади.

Виртуаллик ворисликка ўтади. Функция виртуал деб эълон қилингандан сўнг хосила синфда қайта таърифи(шу прототип билан) бу синфда янги виртуал функцияни яратади, бу холда virtual спецификатори талаб қилинмайди.

Эътибор беринг main усули ичида А синфи ўзгарувчиси таърифланиб, В синфи объекти ёрдамида инициализация қилинган. Кейинги қаторда callme усули чақирилган. Транслятор А синфида callme усули мавжудлигини текширди, бажарувчи тизим ўзгарувчида В объекти сақланганлиги учун, А синфи эмас, В синфи callme усулини чақиради.

Усулларни қайта таърифлаш

Point синфининг янги остки синфи Point3D ўз суперсинфининг distance усулини мерос қилиб олади (мисол PointDist.java). Лекин Point синфида текисликда нуқталар орасида масофа қайтарувчи distance(mt x, int y) усули берилган. Биз бу усулни уч ўлчовли фазога мос келадиган қилиб, қайта таърифлашимиз (**override**) лозим.

final

Хамма усуллар ва ўзгарувчилар кўзда тутилган бўйича қайта таърифланиши мумкин. Агар ворис синфда бирор ўзгарувчи ёки усулни қайта таърифлашга хаққи йўқлиини кўрсатиш учун уларни final (Delphi / C++ тилида **virtual** сўзини ёзмаслик керак) деб таърифлаш лозим.

final int FILE_NEW = 1;

Қабул қилинган қоида бўйичаПо общепринятому соглашению при выборе имен переменных типа final типидаги ўзгарувчиларни номлашда фақат юқори регистрдаги символлардан фойдаланилади. (C++ тилида препроцессор константалар). Баъзида final- усуллардан фойдаланиш код бажарилишини тезлаштиради — чунки, транслятор уларни жойлаштирилувчи (inline) код деб эълон қилади (байт-код тўғридан тўғри кодга жойлаштирилади).

Абстракт синфлар

Жуда бўлмаса битта абстракт усулга эга синф абстракт синф деб аталади. Абстракт усул деб қуйидаги кўринишга эга компонент функцияга айтилади:

abstract<тип> <исм > (< формал_параметрлар_рўйхати>);

Бу синф объектларини яратиш мумкин эмас. Абстракт синф фақат хосила синфлар учун асос синф сифатида ишлатилиши мумкин.

Хар қандай abstract усулга эга синф, abstract деб таърифланиши шарт. Бундай синфларда тўлиқ реализация мавжуд бўлмагани учун, new оператори ёрдамида вакилларини яратиш мумкин эмас. Бундан ташқари абстракт конструкторлар ва статик усуллар эълон қилиш мумкин эмас. Абстракт синф хар қандай вориси ёки суперсинф абстракт усулларини тўлиқ реализация қилиши керак, ёки ўзи абстракт деб элон қилиниши керак.

Глобал Суперсинф

Object: глобал суперсинф

Object синфи хамма синфлар аждоди хисобланади. Java тилида хар бир синф Object синфини кенгайтиради. Лекин class Employee extends Oblects ёзиш шарт эмас. Агар суперсинф ошкор кўрсатилмаган бўлса Object суперсинф хисобланади. Java тилида хар бир синф Object синфини кенгайтиргани учун Object синфи имкониятларини билиш мухимдир.

Object типидаги ўзгарувчини ихтиёрий типдаги объектга илова сифатида ишлатиш мумкин:

Object obj = new Employee("Гарри Хакер", 35000);

Бу типдаги ўзгарувчидан фойдаланиш учун аввал бошланғич типни аниқлаб, типларни келтиришни амалга ошириш лозим:

Employee e = (Employee) obj;

equals ва toString усуллари

Object синфининг equals усули икки объект бир хиллигини текширади. Лекин equals усули Object синфига тегишли бўлгани учун, иккаласи бир хотира қисмига илова қилганлигини текширади. Икки объект эквивалентлигини текшириш учун equals усулини қўшимча юклаш лозим. Мукаммал equals усули яратиш қоидалари.

1. Ошкор otherObject параметрини чақириш — кейинчалик унинг типини other деб аталган бошқа ўзгарувчи типига келтириш лозим.
2. Текшириш, this ва otherObject иловалар бир хилми:

```
if ( this == otherObject) return true;
```

Одатда объектлар майдонини солиштиргандан кўра иловаларни солиштириш осондир.
3. Текшириш otherObject илова нульга (null) тенгми. Агар ха бўлса false қиймат қайтариш. Бу текширишни албатта амалга ошириш лозим.

```
if (otherObject == null) return false;
```

4. Текшириш this ва other объектлари битта синфга тегишлими.
Бу текшириш "симметриклик қоидасига " кўра мажбурийдир.

```
if (getClass() != otherObject.getClass()) return false;
```

5. Талаб қилинган синф ўзгарувчисига otherObject объектини ўзгартириш:

```
ClassName other = (ClassName)otherObject;
```

6. Хамма майдонларни солиштириш. Асосий типдаги майдонлар учун == оператори, объектли майдонлар учун —equals усули қўлланади. Агар икки объект хамма мйдонлари бир хил бўлса true қайтариш, акс холда false.

```
return fieldl == other.fieldl
&& field.2. equals (other . field2)
```

Объект типини getClass усули орқали аниқланади. Объектлар ўзаро тенг бўлиши учун бир синф объектлари бўлишлари керак.

Ворис ичида авввал сперсинф equals усулини чақириш лозим. Агар бу текшириш false қиймат қайтарса, демак объектлар тенг эмас. Агар текшириш муваффақиятли бажарилса остки синф майдонларини текширишга ўтиш мумкин.
Object синфининг яна бир мухим усули toString, бўлиб объектни сатр шаклида қайтаради. Бу усул деярли хамма синфларда қўшимча юкланади, ва объект холатини босмага чиқаришга мўлжалланган.
Кўп (хаммаси эмас) toString усуллари синф номи дан иборат бўлиб, квадрат қавсларда майдонлари қийматлари кўрсатилади.
Албатта ворис синф яратган дастурчи ўз toString усулини яратиши ва ворис синф номини қўшиши лозим. Агар суперсинфда getClass ().getNamef() усули чақирилса, ворис синф super.ToString ( ) усулини чақиради.

Агар объект сатр билан "+" амали ёрдамида конкатенация қилинса Java тили компилятори объект жорий холатини олиш учун автоматик равишда toString усулини чақиради.
Бирорр x — ихтиёрий объект учун дастурчи System.out.println(x) усулини чақирсин; Бу холда println усули x. toString () усулини чақиради ва натижа сатрини чиқаради.Object синфида аниқланган toString усули синф номи ва объект адресини чиқаради. Бунинг сабаби шуки PrintStream синфида toString усули қўшимча юкланмаган.Стандарт библиотекага тегишли кўп синфларда toString усули шундай аниқланганки, унинг ёрдамида дастурни созлаш учун керакли маълумот олиш мумкин. Баъзи созловчилар объектлар холатини экранда акслантинриш учун toString усулини чақиришга имкон беради. Шунинг учун дастур трассировкасида, қуйидаги ифодалардан фойдаланиш мумкин

```
System.out.println{"Жорий холат = " + position);
```

Файллар билан ишлаш

Киритиш чиқариш. Киритиш манбаи деб қурилмаларнинг абстрактлашган қурилмаларнинг тушунчаси ётади. Худди шунингдек чиқариш манбаи деб турли чиқариш қурилмалари тушунилади. Абстрактлашган киритиш чиқариш шуни назарда тутадики дастур кодида киритиш чиқариш қурилмалари конкрет тасвирланмайди. Бу абстракт тушунчалар оқим тушунчаси ёрдамида амалга оширилади. Оқимлар билан ишлаш учун дастурда оқимли синфлардан фойдаланиш зарурдир. Оқимли синфлар абстракт синфлар бўлиб дисклар, ташқи хотира ва тармоқ билан ишлашга имкон беради.

InputStream синфи киритиш учун бу синфдан фойдаланилади. Бу синф қуйидаги усулларга эга:
read усули ва бу усул учта кўринишга эга

- Оқимдан бутун сон ўқийди;
- Массив максимал элементларини ўқийди;
- Берилган байтдан бошлаб берилган сондаги байтларни ўқийди ва берилган массивга ёзиб қўяди.

skip усули. Оқимдан берилган сондаги байтларни ўтказиб юборишга харакат қилади.

available усули ўқиш мумкин бўлган максимал байтлар сонини қайтаради.
close усули киритиш манбаини беркитади. Беркитилган манбадан ўқиш истисно генерация қилинишига олиб келади.

OutputStream бу синф хам абстракт бўлиб маълумотларни чиқариш учун ишлатилади. Усуллари қуйидагича:
write усули уч хил кўринишга эга, улар қуйидагича:

- Берилган сонни чиқариш оқимига ёзади;
- Берилган байтлар массивини тўлалигича чиқаради;
- Байтлар массивини берилган позициядан бошлаб берилган сондаги элементларни оқимга чиқаради.

flash усули чиқариш оқимини тозалайди.
close усули чиқариш оқимини беркитади. Беркитилган оқимга чиқариш хатоликка олиб келади.

Файлли оқимлар :

1. *FileInputStream*-файлдан ўқиш учун ишлатилади, 2та конструкторга эга.
2. *FileOutputStream*-файлга ёзиш учун ишлатилади. Бу синф хам 2та конструкторга эга. Бу синф объекти яратилганда агар файл мавжуд бўлса қайтадан яратилади ва ундаги маълумотлар ўчиб кетади.
3. *ByteArrayInputStream*-бу синфлар берилган байтлар массивини ўқийди.
4. *ByteArrayOutputStream*- берилган байтлар массивини тўлдириш учун ишлатилади. Иккала холда хам ички буфери яратилади. Бу буфер хажми ишлатилган конструкторга қараб 32 ёки 1024 байт бўлиши мумкин.
5. StringBufferInputStream-биринчи синфга ўхшаган бўлиб фарқи шундаки ички буфер объектлари String типига тегишли бўлади.

Файллар билан ишлаш хусусиятлари

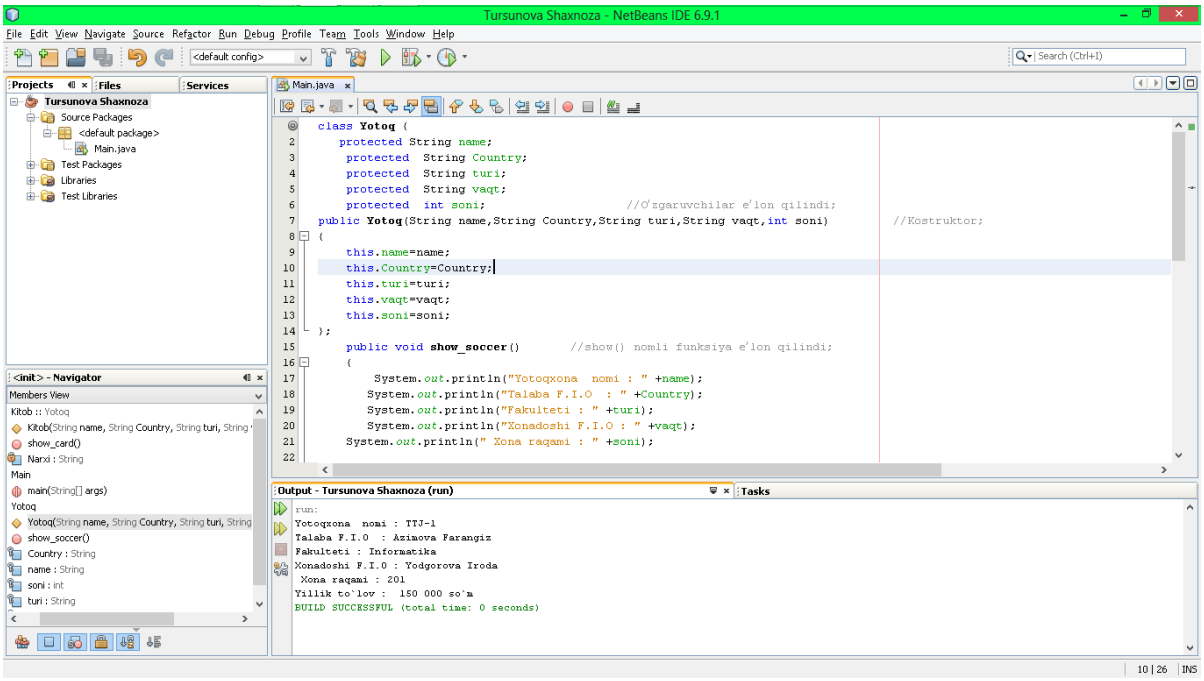
Тўғридан тўғри дискдаги файллар билан ишлаш учун java.io пакетида жойлашган File синфидан фойдаланилади. Java тилида каталоглар оддий файллар сифатида чиқарилади. Оддий файллардан фарқи шундаки каталоглар ўз ичига файллар рўйхатини олади. Бу рўйхатни File синфининг List усули ёрдамида ўқиб олиш мумкин.

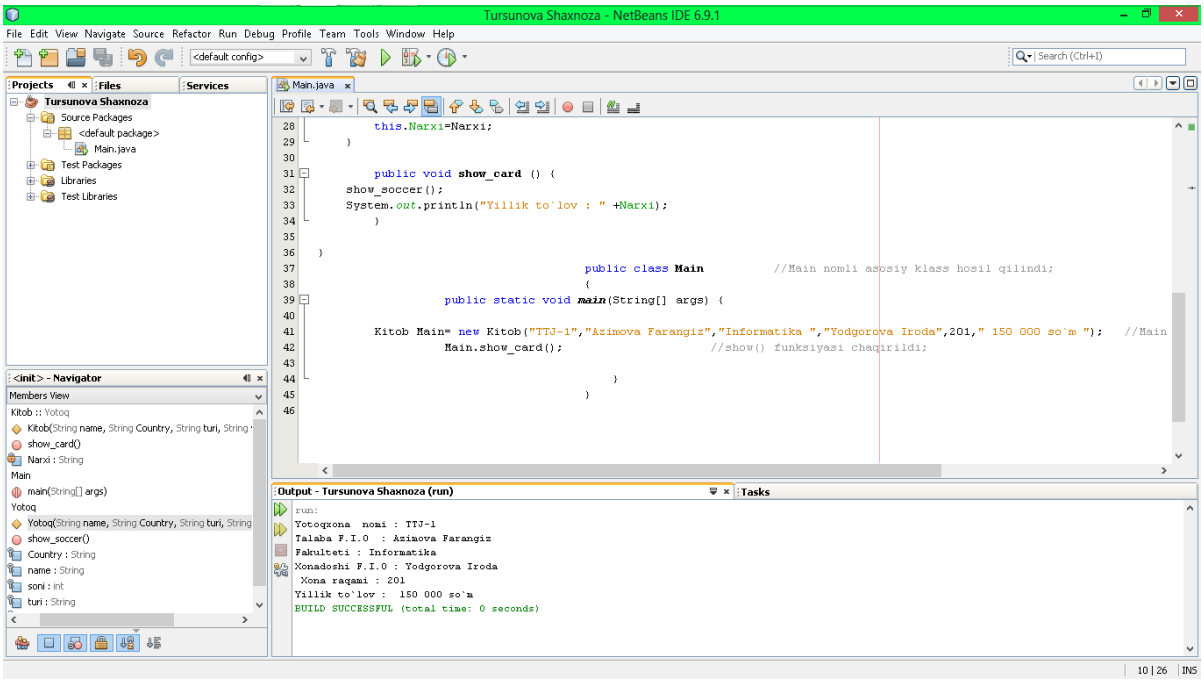
Каталогда файл ва остки каталоглар номлари ва уларга олиб борувчи йўл UNIX ёки MS DOS операцион тизимида қабул қилинган қоидалар асосида ёзилади, лекин Windows мухитида дастур бажарилганда автоматик равишда Windows да қабул қилинган қоидалар асосида қайтадан ёзилади. Агар файллар рўйхати олинганда уларнинг номларига бирор шарт қўйиш лозим бўлса FileNameFilter интерфейсидан фойдаланиш мумкин. Дастурда бу интерфейсдан фойдаланиш учун битта усул accept() усулиниқайта таърифлаш етарлидир. Бу синфнинг юқорида кўрилган синфлардан фарқи шундаки хамма усуллари синхронизация қилингандир, яъни бир нечта жараён бир пайтнинг ўзида ишлаганда турли тасодифий ходисилар юз бериш олди олингандир.

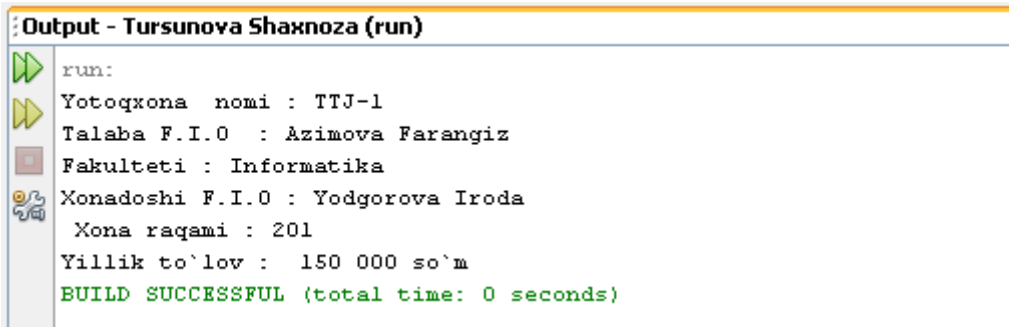
```
class Yotoq {
  protected String name;
   protected  String Country;
   protected  String turi;
   protected  String vaqt;
   protected  int soni;                //O'zgaruvchilar e'lon qilindi;
public Yotoq(String name,String Country,String turi,String
vaqt,int soni)       //Kostruktor;
{
```

```
    this.name=name;
    this.Country=Country;
    this.turi=turi;
    this.vaqt=vaqt;
    this.soni=soni;
};
    public void show_soccer()      //show() nomli funksiya e'lon
qilindi;
    {
        System.out.println("Yotoqxona  nomi : " +name);
       System.out.println("Talaba F.I.O  : " +Country);
       System.out.println("Fakulteti : " +turi);
       System.out.println("Xonadoshi F.I.O : " +vaqt);
    System.out.println(" Xona raqami : " +soni);

} }
class Kitob extends Yotoq {
    private String Narxi;
    public Kitob(String name,String Country,String turi,String
vaqt,int soni,String Narxi){
        super(name,Country,turi,vaqt,soni);
        this.Narxi=Narxi;
    }

        public void show_card () {
    show_soccer();
```

```
    System.out.println("Yillik to`lov : " +Narxi);
      }

}
                    public class Main        //Main nomli asosiy
klass hosil qilindi;
                    {
            public static void main(String[] args) {

      Kitob Main= new Kitob("TTJ-1","Azimova
Farangiz","Informatika ","Yodgorova Iroda",201," 150 000 so`m
");  //Main class o'zgaruvchilariga qiymatlar berildi;
            Main.show_card();              //show() funksiyasi
chaqirildi;

                  }
                }
```







Хулоса

KURS ISHINI BAJARISH DAVOMIDA QUYIDAGI XULOSALARGA EGA BO'LDIM: BULARGA QISM DASTURLARNI YARATISH, OB'EKT XUSUSIYATLARI BILAN

ISHLASH, SINFLARNI SHAKLLANTIRISH KABI TUSHUNCHALARNI O'ZLASHTIRISH .

USHBU KURS ISHIDA YUQORIDA TA'KIDLAB O'TILGAN MA'LUMOTLARNI O'ZLASHTIRISH BILAN BIR QATORDA JAWA VA SHU KABI DASTURLASH TILLARIDA SINFLAR VA OB'EKTLAR, SINFLAR ORASIDAGI MUNOSABATLAR, OB'EKLARGA QIYMATLARNI O'ZLASHTIRISH VA ULARDAN FOYDALANISH KABI TUSHUNCHALAR O'ZLASHTIRILDI.

Фойдаланилган адабиётлар рўйхати

1. Смирнов Н.И. **Java -2.** Учебное пособие. М.:«Три Л», **2000. -320** с.

2. Арнольд К. и др. Язык программирования **JAVA**/ Вильямс. **2001. 624** с.
3. Брюс Эккель. Философия **Java** . Библиотека программиста. **2003.**
4. Грехем И. Объектно ориентированные методы. Принципы и практика. Вильямс., **2004, 879** с.,
5. Иванова Г.С. Объектно ориентированное программирование. Учебник., МГТУ им Баумана, **2003, 320** с.