ЎЗБЕКИСТОН РЕСПУБЛИКАСИ ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

БУХОРО МУҲАНДИСЛИК-ТЕХНОЛОГИЯ ИНСТИТУТИ

“ИНФОРМАТИКА ВА АХБОРОТ ТЕХНОЛОГИЯЛАРИ” КАФЕДРАСИ

“ТИЗИМЛИ ДАСТУРИЙ ТАЪМИНОТ” ФАНИДАН

КУРС ИШИ

Мавзу: _____

_____

Бажарди: **_____ гуруҳ талабаси**

_____

Қабул қилди: **Ибрагимов У.М.**

Бухоро-2013 й.

БУХОРО ЮҚОРИ ТЕХНОЛОГИЯЛАР МУҲАНДИСЛИК-ТЕХНИКА ИНСТИТУТИ

“ТЕХНОЛОГИК ЖАРАЁНЛАРНИ АВТОМАТЛАШТИРИШ” ФАКУЛЬТЕТИ

“ИНФОРМАТИКА ВА АХБОРОТ ТЕХНОЛОГИЯЛАРИ” КАФЕДРАСИ

ТАСДИҚЛАЙМАН
Кафедра мудири
Доц. Раззоқов Ш.И.
«____» _____2013 й.

_____

(Талабанинг гуруҳи ва Фамилияси, исми ҳамда шарифи)

_____

Фанидан

ТОПШИРИҚ

Мавзу: _____
_____
_____

1. **Курс ишининг топшириш учун муддати** _____
2. **Курс ишини бажариш учун бошланғич маълумот** _____

_____

3. **Тушунтириш хатининг мазмуни** _____

_____

4. **График материаллар:** _____
5. **Топшириқ берилган сана:** _____
6. **Топшириқни бажариш учун қабул қилдим:** _____
7. **Рахбар:** _____

БУХОРО – 2013 й.

Мундарижа

Кириш

«Тизимли дастурий таъминот» фани Ўзбекистон Республикаси Президенти И.А. Каримов томонидан қўйилган малакали мутаҳассисларни тайёрлаш миллий таълим тизимини ривожлантириш бўйича муҳим масалаларни ечишда катта ахамият касб этади.

Мазкур фан замонавий электрон ҳисоблаш машиналарининг тизимли дастурий таъминотини ўрганган ҳолда, уни жамиятнинг турли соҳаларида самарали қўллаш муаммолари билан шуғулланади.

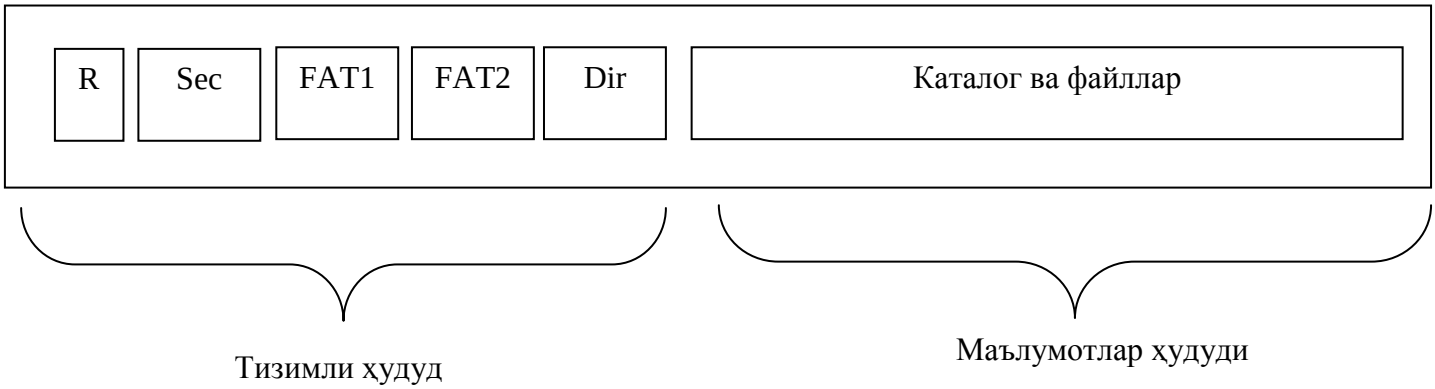
Фан таркибида ҳисоблаш тизимининг дастурий таъминоти, уни ташкил этиш, бошқариш асослари ва ундан амалиётда фойдаланиш масалаларини ўрганиш кўзда тутилган.

Тизимли дастурий таъминот таркибига ҳисоблаш тизимини бошқарувчи дастурлар, хизматчи дастурлар, дастурлаш тизимлари ва операцион тизимлар киради. Ҳозирги вақтга келиб эса, ушбу юқорида келтирилган дастурларни ўз соҳасида қўлламаётган мутаҳассисни топиш мумкин эмас. Демак, ушбу дастурлардан фойдаланиш кўникмасига эга бўлиш нафақат ушбу соҳада хизмат қилаётган мутаҳассислар учун, балки, жамиятимиздаги барча соҳа мутаҳассислари учун зарур ҳисобланади.

Дисклар ва файл тузилиши
FAT файл тизими

Айтиб ўтганимиздек FAT аббревиатураси (file allocation table) “файлларни жойлашиш жадвали” деб айтилади. Бу термин файллар тўғрисида чизиқли жадвал структурага тегишли-файллар номлари, уларнинг аттрибутлари ва боша маълумотлар файлларни жойлашувини аниқлайди (ёки уларни қисмларини) муҳитида. FAT элементи физик файл бошланиши жойлаган дискнинг ҳудудини аниқлайди.

FAT файл тизимида ихтиёрий мантиқий дискнинг дискли ҳудуди иккита қисмга ажралади: тизимли ва маълумотлар ҳудуди.



Мантиқий дискнинг тизимли ҳудуди форматлаш жараёнида яратилади,кейинчалик файл структурасини манипуляциялашда ўзгаради. Мантиқий диск маълумотлар ҳудуди файл ва каталогларни сақлаб илдиз каталогига бўйсинади. У тизимли ҳудуддан фарқли равишда фойдаланувчи интерфейси учун мурожаатли. Тизимли ҳудуд кетма-кет мантиқий адресли кенгликка жойлашган компонентлардан ибоарт.

-юкланувчи ёзув (boot record, BR);
-банд қилинган секторлар (reserved sector, ResSec);
-файллар жойлашган жадвали (file allocation table, FAT);
-илдиз каталог (root directory, RDir).

Файллар жойлашиш жадвали

Файлларни жойлашиш жадвали муҳим информацион структура ҳисобланади. Айтиш мумкинки, у маълумотлар ҳудуди харитасини ифодалайди, унда маълумотлар ҳудуди ҳар бир қатнашчиси ҳолати изоҳланган. Маълумотлар ҳудуди кластерлар – деб номланувчи бўлакларга бўлинган. Кластер бу адресли мантиқий диск кенглигида кетма-кет бир ёки бир неча секторлар бирлашмаси ҳисобланади. FAT жадвали бир файлга тегишли кластерлар занжиридек боғланади.

Кластер- файлга ажратилган энг кичик адресланган дискли хотира бирлиги. Файл ёки еаталог бутун кластер сонин эгаллайди. Охирги кластер бунда тўлиқ ишлатилмаслиги мумкин, бу эса кластерларни ҳажми катта бўлганда диск кенглигини ортиқча сарфлашга олиб келади. Дискеталарда кластер 1 ёки 2 секторни, қаттиқ дискларда ҳажмига қараб эгаллайди.

FAT ва VFAT файл тизимларининг асосий камчилиги мантиқий дискни катта ҳажмларида кластерлашда катта сарфни юзага келиши ҳисобланади ва мантиқий дискнинг ҳажмига чегараланиш бор. Бу эса ўша FAT жадвали ғоясини ишлатган ҳолда янги файл тизмини қўллашга олиб келди. Шунинг учун Microsoft Windows 95 OEM Service 2 да VFAT ўрнига FAT32 файл тизими келди. FAT32 тўлиқ мустақил 32-разрядли файл тизими ҳисобланади ва кўпгина мукаммаллик ва қўшимчаларга эга.

Ринципиал фарқи шундаки, FAT32 диск кенглигини эффективроқ сарфлайди. Биринчи навбатда FAT32 тизими олдинги версияларидаги файл тизимларига нисбатан кичик

ҳажмдаги кластерларни ишлатади, қайсики улар 65535 кластер билан чегараланар эди (диск ҳажми ошиши билан кластерлар ҳажмини ошишига тўғри келар эди). Шундай экан 8Гб гача ҳажмдаги дисклар учун FAT32 4Кб кластерларни ишлатиши мумкин.
FAT32 илдиз каталоги жойини ўзгариши ва FAT нинг нусхасини резервлаши мумкин. FAT32 нинг илдиз каталоги оддий кластерлар занжири сифатида ифодаланган. Шундай экан, илдиз каталог дискнинг ихтиёрий жойида бўлиши мумкин.
FAT сиғимини 4Еб оширишидан ташқари, FAT32 файл тизими бошқа ўзгаришларни ҳам илдиз каталогига амалга оширди. Олдинги қўлланишлар илдиз каталог дискда бир кластерда жойлашишини талаб этарди. Бунда илдиз каталог 512 да нортиқ файлларни сақла йолмасди.

NTFS

NTFS файл тизими номида “New Technology” сўзи мавжуд, яъни “янги технология”. Ҳақиқатда, NTFS бир қатор ўзгаришлар ва янгиланишларга эга бўлиб, уни бошқа файл тизимларидан ажралиб туришини таъминлайди. Фойдаланувчилар нуқтаи назаридан файллар олдингидек каталогларда (Windows да кўпинча “папка” ёки “фолдер” деб айтилади) сақланади. Аммо NTFS да FAT дан фарқли равишда катта ҳажмдаги дискларда ишлар эффективроқ юз беради; каталог ва файлларга мурожаатни ташкил этиш учун воситаларга эга; файл тизими чидамлилигини оширувчи механизмлар киритилган; диск секторлар ва кластерлар максимал сонига бўлган кўпгина чегараланишлар олиб ташланди.

NTFS файл тизимининг асосий имкониятлари

-чидамлилик. Юқори кўрсаткичли компьютерлар ва замонавий фойдаланиш тизимлари (серверлар) юқори чидамлиликка эга бўлиши керак, бу эса NTFS нинг ўзини тутиши ва структурасида муҳим элемент ҳисобланади. Чидамлилик оширишнинг усулларидан бири транзакциялар механизмининг қўллаш ҳисобланиб, файллар устидаги операциялар жцрналлаштириш амалга оширилади;
-кенгайтирилган функционаллик. NTFS кенгайиши ҳисобга олинган ҳолда қурилган. Унга қўшимча имкониятлар киритилган-мукаммаллаштирилган барқарорлик, бошқа тизимларнги эмуляциялаш, кучли хавфсизлик модели, маълумотлар оқимини паралелл қайта ишлаш ва фойдаланувчи томонидан аниқланган файл атрибутларини яратиш;
-POSIX ни қўллаб-қувватлаш. АҚШ ҳокимияти улар томонидан сотиб олинадиган барча тизимлар минимал даражада POSIX мос стандартини қўллаб-қувватлашини талаб этар экан, бундай имконият NTFS да ҳисобга олинган. POSIX файл тизими асосий воситаларидан бири файл номлари ҳарфларни катта кичиклигига шарт бўлган талаб, файлга охирги маротаба мурожаат вақти ва “қаттиқ мурожаатлар” юеб номланувчи механизм- альтернатив номлар, яъни юир файлга бир ёки бир неча ном билан мурожаат этиш имкониятларининг қўшилиши тушунилади.
-эгилувчанлик. Диск ҳудудининг тақсимланган модели NTFS да эгилувчанлиги билан фарқ қилади. Кластер ҳажми 512 байтдан то 64 Кб ўзгариши мумкин, у диск фазоси тақсимотининг ички квантига каррали сонни ифодалайди.

NTFS файллар узун номланиши; UNICODE кодлаш тизимини ва эски FAT номлари (8.3) қўллаб-қувватлайди.

NTFS катта маълумотлар массивини қайта ишлаш амалини жуда яхши амалга оширади ва 300-400 Мб ва ундан катта ҳажмдаги бўлимлар билан ишлашда ўзини яхши кўрсатади. Бўлимнинг максимал қиймати (файл ҳажми) 16 Эбайт бўлиши мумкин. Илдиз ва илдиз бўлмаган каталогда файллар сони чегараланмаган. NTFS каталоглари

структураси асосида “бинар дарахт” деб номланувчи эффектив маълумотлар структураси ётар экан, NTFS да файлларни қидириш вақти файллар сони билан боғлиқ эмас.

NTFS тизими ўзини тиклаш воситасига эга. NTFS тизим яхлитлигини текшириш турли механизмларини қўллаб-қуватлайди, хусусан транзакциялар журнали, у эса махсус тизимли журналдан файлли операцияларни ўқишга имкон беради.

NTFS файл тизими NT хавфсизлик объектли моделини қўллаб-қувватлайди ва барчабўлимлар, каталоглар ва файлларни алоҳида объект сифатида қарайди. NTFS файллар даражаси ҳавфсизликни таъминлайди; бу эса бўлим, каталог ва файлларга мурожаат ҳуқуқи фойдаланувчи ва улар қайсми гуруҳга таъаллуқлигига боғлиқ. Файл тизимини объектига мурожаат ҳуқуқини текширади. Агар фойдаланувчи етарлича ҳуқуқ даражасига эга бўлса унинг сўрови қўллаб-қувватланади, акс ҳолда сўров рад этилади. Бу хавфсизлик модели NT компьютерларига фойдаланувчиларни маҳаллий қайд этишда ҳам, узоқлашган тармоқли сўровларда ҳам қўлланилади.

Ва ниҳоят, катта ҳажмдаги бўлимлар ва файллардан ташқари NTFS қуйилган сиқиш воситаларига эга, қайсики уларни алоҳида файлларга, каталогларга ва бўлимларга қўллаш мумкин.

NTFS файл тизими бўлим структураси

файл тизими структурасини кўриб ўтамиз. Бу ерда биз унинг асосий қисмларини кўриб ўтамиз.

NTFS билан ишлаганда ишлатиладиган асосий тушунчалардан бири бўлим (volume) ҳисобланади. Бир неча бўлимлардан иборат таъсирларга чидамли бўлимларни яратиш имконияти, яъни RAID – технологияни ишлатиш. Бошқа тизимлар сингари NTFS ҳам диск бўлими ҳудудини кластерларга – ммаълумотлар бирлиги сифатида адресланади бўлади. NTFS кластерлар ҳажми 512 байтдан то 64 Кб бўлган кластерни ташкил қўллаб-қувватлайди, стандарт ҳолатда кластер ҳажми 2 ёки 4 Кб бўлади.

Диск ҳудудуининг бутун ҳудудини NTFS иккита тенг эмас қисмга ажратади. Биринчи 12% и VFT ҳудуд деб номланадиган ҳудудга ажратади, қайсики бу ҳудуд жой эгаллаши, ҳажми ўзгариши мумкин, асоссий метафайлни MFT 2. Бу ҳудудга бирон-бир маълумотларни ёзиш мумкин эмас. MFT ҳудуд доим бўш ҳолатда сақланади- бу энг асосий файл (MFT хизматчи файл) ҳажми ошганда фрагментланмаслиги учун қилинади. Қолган 88% и файлларни сақлаш учун оддий ҳудуд бўлиб хизмат қилади.

MFT	MFT ҳудуди	Файл ва каталогларни жойлаштириш учун ҳудуд	MFT нинг 1-16 та ёзуви нусхаси	Файл ва каталогларни жойлаштириш учун ҳудуд

MFT (master file table, файлларнинг умумий жадвали) дискнинг барча қолган файлларини (ўзини ҳам) марказлаштирувчи каталоги сифатида ифодаланади. MFT аниқ ҳажмли (1 Кб) ёзувларгабўлинган ва ҳар бир ёзув бирон бир файлга мос келади. Биринчи 16 файл (ёзув) хизматчи характерга эга ва Операцион тизимга мурожаатли эмас – улар метафайллар деб айтилади, бунда биринчи метафайл MFT нинг ўзи. бу 16 элементи аниқ ўринга эга дискнинг ягона қисми ҳисобланади. Бу 16 ёзувнинг 2-нусхаси тизим барқарорлиги учун бўлимнинг ўртасида жойлашади, чунки улар жудамуҳим ҳисобланади. MFT файлларнинг қолган қисмлари бошқа файллар сингари дискнинг ихтиёрий қисмида жойлашиши мумкин – унинг жойлашувини ўзига “ёпишиб MFT нинг 1-элементига тиклаш мумкин.

MFT метафайллари

Метафайл номи	**Метафайл вазифаси**
$MFT	Master File Table ўзи

$MTmirr	Бўлим ўртасида жойлаштирилган нинг биринчи 16 ёзуви нусхаси
$LogFile	Операцияларни журл\наллашни қўллаб-қувватловчи файл
$Volume	Хизматчи информация – бўлим белгиси, файл тизими версияси ва ҳоказо
$AttrDef	Бўлимдаги файлларнинг стандарт аттрибутлари рўйхати
$	Илдиз каталог
$BitMap	Бўлим бўш ўрни харитаси
$Boot	Юкланувчи сектор (агар бўлим юкланадиган бўлса)
$Quota	Диск ҳудудуини ишлатиш бўйича фойдаланувчилар ҳуқуқи ёзилган файл (бу файл Windows 2000 да NTFS 5.0 версиясида пайдо бўлди)
$Upcase	Файл – файл номларида мос равишда катта ва кичик ҳарфларни жадвали. NTFS да файл номлари Unicode да сақланади (бу эса 65 минг турли рамзларни ташкил этади)

Айтиб ўтилган MFT нинг 16 та файли (метафайллар) хизматчи характерга эга; уларнинг ҳар бири тизимнинг бирон бир ишига жавоб беради. Метафайллар MFT бўлими илдиз каталогида жойлашган. Уларнинг барчаси номи «$» белгисидан бошланади, лекин улар тўғрисида маълумот олиш учун стандарт усуллар етарли эмас.

Юқорида келтирилган жадвалда асосий маълум метафайллар вазифаси келтирилган. Шундай қилиб масалан $MFT файли ҳажмини кўриб бўлимини каталоглаштиришда операцион тизим қанча вақт сарфлашини кўриш мумкин.

Демак, бўлимнинг барча файллари MFT да таъкидланади. Бу структурада файллар ичидаги маълумотлардан ташқари файллар тўғрисида барча информация сақланади. Файл номи, ҳажми, алоҳида фрагментларни дискдаги жойи ва ҳоказо – буларнинг барчаси мос ёзувларда сақланади. Агар информация учун MFT нинг битта ёзуви етмаса, унда бир неча ёзув шлатилади ва бунда албатта кетма-кет бўлиши шарт эмас. Файллар катта ҳажмда бўлмаслиги мумкин. Унда яхши ечим қўлланилади: файл маълумотлари MFT да сақланади. Юз байтли файллар асосан асосий “физик” ҳудудига эга эмас – бундай файлларнинг маълумотлари бирта жойда сақланади, яъни MFT да.

NTFS да ҳар бир файл оқимлари (streams) ёрдамида ифодаланган, яъни унинг “оддий маълумотлари” йўқ “оқимлар” бор. Оқимни тўғри тушуниш учун оқимлардан бири файл маълумотлари маъносини қилиб, файлнинг асосий предмети ягона - MFT даги номер, қолганлар эсаоқимлари ҳам . Бундай услуб эффектив фойдаланишга ёрдам беради, масалан, қолган яна битта оқимни елимлаш мумкин ва у орқали унга ихтиёрий маълумотларни ёзиш мумкин. Қизиғи шундаки, бу қўшимча оқимлар файллар билан ишлашнинг стандарт воситаларидан кўриб бўлмайди: файл ҳажмини кузатиш-бу фақат асосий оқимнинг ҳажми ҳисобланади, қайсики у анъанавий маълумотларни сайлайди. Масалан 0 узунликдаги файлга эга бўлиш мумкин, уни ўчирганда эса 1 Гб бўш ўрин пайдо бўлади чунки, бирон-бир доно дастур ёки технология унга шундай катта ҳажмли қўшимча оқимни (альтернатив маълумотлар) “улимлаган”. Аслида ҳозирги кунда оқимлар ишлатилмайди, шу учун бундай ҳолатлардан қўрқиш керак эмас, лекин назарий жиҳатдан шундай бўлиши мумкин. Фақат шуни тушуниш керакки, NTFS да файл - бу чуқурроқ тушунча ҳисобланади.

NTFS бўлимида каталог ва файллар учун стандарт аттрибутлар аниқ ном ва код типига эга ва улар қуйидаги жадвалда келтирилиган

Тизимли аттрибут	**Аттрибут изоҳи**
Файл тўғрисида стандарт	Анъанавий аттрибутлар read only, hidden, archive, system, вақт белгиланишлар, яратиш вақти ёки охирги ўзгартириш вақти,

информация	файлга мурожаатловчи каталоглар сони ва ҳоказо
Аттрибутлар рўйхати	Файл ташкил топган аттрибутлар рўйхати ва MFT ёзувга файлли мурожаат, қайсики MFT да файл аттрибутлари жойлашган. Охиргиси агар файл учун MFT да бир неча ёзув ишлатилса керак бўлади
Файл номи	Unicode да файл номи. Файл бир неча аттрибутга эга бўлиши мумкин.
Ҳимоя дескриптори	Файлни ноқонуний мурожаатдан сақловчи маълумотларни ҳимоялаш структураси (ACL). “Ҳимоя дескриптори” аттрибути файл эгаси ким ва унга ким мурожаат қилиб билишини аниқлайди
Маълумотлар	Файл маълумотларининг ўзи, унинг таркиби. NTFS да файлнинг жимлик қоидаси бўйича битта номсиз аттрибути бор у қўшимча номли маълумотлар аттирбутига эга бўлиши мумкин. Каталогни жимлик қоидаси бўйича маълумотлар аттирбути йўқ, лекин у мажбур бўлмаган номланган маълумотлар аттрибутига эга бўлиши мумкин.
Индекс илдизи, индексни жойлашуви, битли харита (фақат каталоглар учун)	Катта каталогларда файл номлар индекси учун ишлатиладиган аттрибутлар
кенгайтирилган аттрибутлари	Windows NT файл-серверида OS/2 ички тизимлари ва OS/2 мижози учун HPFS кенгайтирилган аттирбутларини амалга ошириш учун қўлланиладиган аттрибут

MFT ёзувларида файл аттрибутлари код типлари сонли қийматлари ўсиш тартибида жойлашган, бунда баъзи аттрибутлар ёзувда бир неча маротаба учраши мумкин: масалан агар файлнинг маълумотлар аттрибути бир нечта бўлса ёки бир нечта номи бўлса.

NTFS да файл номи FAT ва HPFS файлларига нисбатан миллий алифболарда киритилган ихтиёрий рамзларни сақланиши мумкин, чунки улар Unicode да ифодаланади – 16 битли кўринишда, у эса 65535 та турли рамзларни ифодалайди. NTFS да файл номига 255 тагача рамзни қабул қилиш мумкин.

Файл тизими эффективлигини каталогларни ташкил этиш катта рол ўйнайди. NTFS да каталог махсус файлни ифодалаб, ўзида бошқа файл ва катлогларга мурожаатларни сайлайди ва дискда иерахик структурани қуради. Файл каталоги блокларга бўлинган, ҳар бир блок файл номини, асосли аттибутларини ва MFT элементига мурожжатни сақлайди, қайсики у каталог элементи тўғрисида тўлиқ информация беради. Дискнинг асосий каталоги – илдизли-оддий каталоглардан фарқи йўқ, фақат MFT метафайлдан унинг бошига махсус мурожаатдан бошқа.

Каталогнинг ички тузилиши бинар дарах ҳолатида, худди HPFS файл тизимига ўхшаб. Айтганча NTFS файл тизимини яратишда HPFS файл тизими кўпгина ютуқларини максимал тарзда қўлланилган. Афсуски бутун дискни зоналарга бўлиш қўлланилмаган, унда бўш кластерлар тўғрисида информация туради. Натижада бу қарашдан фойдаланилмаслик ва транзакцияни қўллаш HPFS га нисбатан NTFS файл тизими ишини секинлаштиради.

Демак, энди бизага маълумки, каталогнинг бинар дарахти файллар номни шундай жойлаштирадики, файл номи икки қийматли жавоб кўринишида қидирилиб файл

жойлашуви тўғрисида саволга жавоб қайтарилади. Бинар дарахт қуйидаги саволга жавоб беради: шу элементга нисбатан қидирилаётган исм қайси гуруҳдалигини яъни тепада ёки пастда? Бу саволни ўртадаги элементдан сўрашни бошлайди ва ҳар бир жавоб қидириш зонасини ўртача икки маротаба қисқартиради. Агар файллар алфавит тартиби бўйича сараланган бўлса деб тасавур қилмак, унда саволга жавоб оддий усулда жавоб қайтрилади – бошланғич ҳарфларни таққослаш орқали. Қидириш ҳудуди икки маротаба қисқаради, яъни қидирув ўртадаги элементдан бошланибдавом этади.

Айтиш керакки, каталогга файлни қўшиш дарахт кўринишида жуда қийин эмас, FAT чизиқли каталог тизимига нисбатан. Бу таққослаш бўйича вақтнинг бир хил кетишига олиб келади. Янги файлни каталога қўшиш учун у ерда шу номли файл йўқлигига амин бўлиш керак. Шунинг учун чизиқли каталогларга ташкил этилган FAT тизимида нафақат файларни қидиришда муаммолар пайдо бўлади.

NTFS файл тизимининг файл ва катлогларга мурожаат чегараларини ўрнатиш бўйича имкониятлари

Файл ва каталогларга турли мурожаат ҳуқуқни ташкил этиш билан боғлиқ асосий имкониятларини кўриб ўтамиз. Бу амаллар файл ва папкани тармоқда мурожаат этиш автоном ишлашда кўриб ўтиш мумкин. NTFS каталогларни (папкаларни) ва файлларни турли объект деб қарайди ва ҳар бир объект типи учун мурожаат ҳуқуқи рўйхатини этади. Қуйида NTFS нинг папкаларга берилган ҳуқуқлари келтирилган (пастда файл учун мос ҳуқуқлар келтирилган):

-мурожжат йўқ (no access) (None) (йўқ);
-тўлиқ мурожаат (full control) (All) (All) (барча) (барча);
-ўқиш ҳуқуқи (read) (RX) (RX) (ўқиш) (ўқиш);
-қўшиш ҳуқуқи (add) (WX) (not specified) (ёзиш/бажариш кўрсатилмаган);
-қўшиш ва ўқиш ҳуқуқи (add&read) (RWX) (RX) (ўқиш/ёзиш/бажариш) (ўқиш/бажариш);
-қўшиш ҳқуқи (list) (RX) (not specified) (ўқиш/бажариш) (кўрсатилмаган);
-ўзгартириш ҳуқуқи (change) (RWXD) (RWXD) (ўқиш/ёзиш/бажариш/ўқириш) (ўқиш/ёзиш/бажариш/ўқириш);

Мурожаат ҳуқуқи номидан сўнг кўрсатилган қавсдаги иккита гапга эътиборқилинг. Биринчи ифода папканинг ўзига тегишли, иккинчиси эса – унинг ичига яратилиши мумкин бўлган барча файлларга тегишли. Масалан, папка учун тўлиқ мурожаат ҳуқуқида ихтиёрий ҳаракатлар рухсат берилади, аммо папкага тўлиқ мурожаат ҳуқуқига эга фойдаланувчи папкада яратилган барча файлларга тўлиқ мурожаат ҳуқуқига эга бўлади (агар файлга мурожаат ҳуқуқлари унинг эгаси ёки администратор томонидан ўзгартирилмаган бўлса). Бошқача айтганда, NTFS да файл ва папкалар жимлик қоидаси бўйича ота-она папка учун ўрнатилган мурожаат ҳуқуқларини мерос қолдиради, аммо бу ҳуқуқлар NTFS мос объектлари учун мурожаат ҳуқуқларини ўзгартиришга эга ихтиёрий фойдаланувчи томонидан ўзгартирилиши мумкин.

NTFS файллар қуйидаги ҳуқуқларга эга бўлиши мумкин:

-тўлиқ мурожаат (full control) (All) (барча);
-мурожаат йўқ (no access) (None) (йўқ);
-ўзгартириш ҳуқуқи (change) (RWXD) (ўқиш/ёзиш/бажариш/ўқириш);
-ўқиш ҳуқуқи (read) (RX) (ўқиш/бажариш)/

NTFS мурожаат ҳуқуқлари учун умумий каталог ҳуқуқлари сингари қамраб олиш қоидаси тасир этади. “Мурожаат йўқ” ҳуқуқи бундан истисно.

Фойдаланувчиларни тармоқдан уланишларида NTFS ҳуқуқлари умумий каталоглар ҳуқуқи билан қарама-қаршиликка дуч келиб қолиши мумкин. Бундай ҳолатда энг қаттиқ қўл чегараланиш мрожаат ҳуқуқи қўлланилади. Кўпчиликда тармоқдан мурожаат

чегараларини тушунишга муаммолар юзага келади. Аммо бунда осон тушуниб олиш мумкин, агар NTFS бўлимларида жойлашган каталог ва файлларга тармоқдан мурожаат пайтида бизда иккита кетма-кет механизмлар таъсир этишини эсласак. Биринчи маротаба тармоқ механизми томонидан аниқланган файлга мурожатга эга бўламиз. Бу “мурожаат йўқ” – “no access” ҳуқуқи, - “full control” ҳуқуқи. Шундан сўнг NTFS хусусияти аниқланган файл ва папкаларга чегараланишлар кучга киради. Яъни иккита тўсиқ кетма-кетлигини ўтишимиз керак. Бошқача айтганда, файл ва папкаларга якуний мурожаат ҳуқуқлар иккала механизмда берилган аксимал чегаралар билан аниқланади.

Юқорида келтирилган ҳуқуқлардан ташқари махсус мурожаат (Special Access) деб номланувчи ҳуқуқ мавжуд. Агар бу мурожаат ҳуқуқини танласак, унда аслида қуйидаги рўйхатдан бир вақтда бир неча ҳуқуқларни танлаш имконияти мавжуд бўлади:

-тўлиқ мурожаат (full control) (All);
-ўқиш (read) (R);
-ёзиш (write) (W);
-бажарилиш (execute) (x);
-ўчириш (delete) (D);
-рухсатни ўзгартириш (change permissions) (P);
-эгасини ўзгартириш (take ownership) (O).

Юқорида кўрсатилган рухсатларнинг хтиёрий тўпламини танлаш мумкин, аммо амалиёта бу ишламайди. Масалан, x (бажариш) ҳуқуқини R (ўқиш) ҳуқуқисиз кўрсатиб бўлмайди, лекин файлларни бошқариш тизимиларида бундай ҳуқуқни амалга ошириш мумкин.

Эслатиб ўтамиз, Windows NT convert.exe деб номланувчи махсус утилитага эга бўлиб, қайсики у FAT бўлимини NTFS бўлимига ўтказади, аммо тескари ўткаўзувчи (NTFS дан FAT га) утилита мавжуд эмас. Бундай тескари ўтказишни амалга ошириш учун, сиз FAT бўлимини очиб NTFS бўлимидан файлларни нусхалаб сўнгра оригиналини зчиришингиз керак. Нусхалашда NTFS дан FAT га шуни эсдан чиқармаслик кеаркки NTFS хавфсизлик аттрибутларининг йўқолиб кетади.

Амалий қисм

```
program leks;
label 10,20,30,40,50;
const
 tts:array[1..20] of string[7]=
 ('program','input','output','var','end','begin','while','do','char',
 ':',',',':=','.','<','<>','(',';','integer','>','=' );
 r:set of char=[':','<','>', ';', ',' ,'.','=','(',')',' ',"",'+','-'];
 c:set of char=['0','1','2','3','4','5','6','7','8','9'];
type ms=array[1..100]of string[10];
  str=string[15];
var
 f,f1:text;ff:file of byte;
 tl ,tsi:ms; kod:array[1..100,1..2]of byte;
  nm,k,l,t,i,j:byte;p:array[1..10]of byte;
  s,h,h1:integer;
 sl,name:str;st:string[100];
PROCEDURE poisk(s:str;lim,nt:byte;var nom:byte);
 var z:integer;
 begin nom:=0;
 if nt=1 then
  for z:=1 to lim do if tts[z]=s then nom:=z;
 if nt=2 then for z:=1 to lim do if tsi[z]=s then nom:=z;
 if nt=3 then for z:=1 to lim do if tl[z]=s then nom:=z;
 end;
PROCEDURE zapis(nt,nom:byte);
 begin
  kod[k,1]:=nt;kod[k,2]:=nom;
  inc(k);sl:='';
 end;
PROCEDURE err(k:byte);
        var str:string;
        begin
        case k of
        1:str:='®вбгвбвўгЃв ;';
        2:str:='®вбгвбвўгЃв ,';
        3:str:='®вбгвбвўгЃв .';
        4:str:='Є®-Ґж д ©« ';
        5:str:='ЁбЇ.-Ґ бЁ¬ў.Ё¬п';
        6:str:='®вбгвбвўгЃв var';
        7:str:='®вбгвбвўгЃв :';
        8:str:='ЇЃаЃ¬Ґ--лЃ -Ґ вЁЇ  integer';
        9:str:='®вбгвбвўгЃв begin';
        10:str:='®иЁЎЎЄ ';
        11:str:='ЇЃаЃ¬Ґ-- п ®ЇЁб -  2 а § ';
        12:str:='®вбгвбвўгЃв do';
        13:str:='®вбгвбвўгЃв §- Є ба ў-Ґ-Ёп';
        14:str:='овбгвбвўгЃв:=';
```

```
        15:str:='ЇҐaҐ¬Ґ-- п -Ґ ®ЇЁб - ';
        end;
        writeln(str);
        halt;
        end;
PROCEDURE rd;
  begin
   if eof(ff) then err(4)
   else read(ff,i,j);
  end;
PROCEDURE pr;
var l:integer;
        begin
        rd;
        if (i<>1)and(j<>12) then err(14);
        rd;
        if (i<>2)and(i<>3)then err(5);
        l:=0;
        for h:=1 to s do
        if j=p[h] then l:=1;
        if l=0 then err(15);
        rd;
        if (i<>1)and(j<>17) then err(1)
                        else rd;
         end;
PROCEDURE op;
    var l:integer;
    begin
    l:=0;
    for h:=1 to s do
    if j=p[h] then l:=1;
    if l=0 then err(15);
    end;
BEGIN
writeln('ўў. Ё¬п д ©« ');
readln(name);
assign(f,name);
reset(f);
writeln('ўўҐ¤ЁвҐ Ё¬п д ©«  ¤«п аҐ§г«мв в®ў');
readln(name);
assign(ff,name);rewrite(ff);
l:=0;k:=1;t:=0;sl:='';
 while not eof(f) do  BEGIN
 readln(f,st);
 i:=1;
 sl:='';
20: while (i<=length(st))and (not (st[i] in r)) do begin
  sl:=sl+st[i];inc(i);end;
```

```
  if st[i]="" then begin
  if sl<>'' then begin
     poisk(sl,20,1,nm);if nm<>0 then zapis(1,nm)
                  else begin poisk(sl,t,2,nm);
                  if nm<>0 then zapis(2,nm)
                  else begin inc(i);tsi[i]:=sl;zapis(2,t);
                  end;end;end;
poisk("",20,1,nm);zapis(1,nm);inc(i);
   while st[i]<>"" do begin
   sl:=sl+st[i];inc(i) end;
   if length(sl)>10 then begin
writeln(f1,'Ћ□ЄЃЉЋ: , ®Ўa Ў влў Ґ¬®¬ д ©«Ґ ¤«Ё-  бЁ¬ў®«м-®© Є®-бв -вл');
writeln(f1,sl,' Ў®«миҐ 10');close(ff);goto 40;end;
   poisk(sl,l,3,nm);
   if nm<>0 then zapis(3,nm)
    else begin inc(l);tl[l]:=sl;zapis(3,l);end;
 poisk("",20,1,nm);zapis(1,nm);inc(i);goto 30;
  end;
  if (st[i] in r) and (sl='') then goto 30;
  if sl[1] in c then begin poisk(sl,l,3,nm);if nm<>0 then zapis(3,nm)
  else begin inc(l);tl[l]:=sl;zapis(3,l);end;goto 30;end;
  poisk(sl,20,1,nm);
  if nm<>0 then zapis(1,nm)
   else begin poisk(sl,t,2,nm);
    if nm<>0 then zapis(2,nm)
    else begin inc(t);tsi[t]:=sl;zapis(2,t);end;end;
30: if i>length(st) then goto 10;
    if st[i] in r then begin
    if (st[i]=':') and (st[i+1]='=')then
                  begin sl:=':=';i:=i+2;goto 50;end;
    if (st[i]='<') and (st[i+1]='>')then
                  begin sl:='<>';i:=i+2;goto 50; end
    else begin
         if st[i]<>' 'then begin sl:=st[i];inc(i);end
         else begin inc(i);sl:='';goto 20; end;end;
       50:poisk(sl,20,1,nm);
          if nm<>0 then zapis(1,nm)
          else begin poisk(sl,t,2,nm);
          if nm<>0 then zapis(2,nm)
          else begin inc(t);tsi[t]:=sl;zapis(2,t);end;end;
    if i>length(st) then goto 10 else goto 20;end;
10:END;
close(f);
   for i:=1 to k-1 do begin
  if i<>k-1 then
  write(ff,kod[i,1],kod[i,2])
  else write(ff,kod[i,1],kod[i,2]);
 end;
```

```
 close(ff);reset(ff);
 rd;
if (i=1)and(j=1) then begin rd;
   if i=2 then begin rd;
      if (i=1)and(j=17) then begin end
         else err(1);end
   else err(5);rd;end;
s:=0;
for h:=1 to 10 do
p[h]:=0;
if (i<>1)and(j<>4) then err(6);
 repeat
 rd;
 if i<>2 then err(5);
 inc(s);p[s]:=j;
 for h:=1 to s do begin
 for h1:=h+1 to s do
 if p[h]=p[h1] then err(11);
 end;
 rd;
until((i<>1)or(j<>11));
 if (i<>1)or(j<>10) then err(7);
rd;
 if (i<>1)or(j<>18) then err(8);
rd;
 if (i<>1)or(j<>17) then err(1);
rd;
 if (i<>1)or(j<>6) then err(9);
rd;
while (i<>1)or(j<>5) do
begin
  if i=2 then begin op;pr;end;
  if (i=1)and(j=7) then begin rd;
            if (i=2)or(i=3) then begin op;rd;end
               else err(5);
            if (i=1)and((j=15)or(j=20)or(j=19)or(j=14)) then rd
               else err(13);
            if (i=2)or(i=3) then begin op;rd;end
               else err(5);
            if (i=1)and(j=8) then rd
              else err(12);
            if (i=2)or(i=3) then begin op;pr;end;
       end;end;
       rd;
       if (i<>1)and(j<>13) then err(3)
       else writeln('ЋиЁЎЄЁ -Ґ - ©¤Ґ-л');
40:end.
```

Хулоса

Ушбу “ мавзу” курс ишини бажариб Тизимли дастурий таъминоти фанидан катта таъсуротларга ва натижаларга эришдим.

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. Браун С. Операционная система UNIX - М.: Мир. 1986.-463 с.
5. Вендров А.М. Case - технология. Современные методы и средства проектирования информационных систем. -М: Финансы и статистика, 1998 -361с
6. Вирт И. Алгоритмы и структуры данных - М.; МИР. 1989. - 360 с.
7. Гордеев А.З., Молчанов А.Д. Системное программное обеспечение. -Спб-Петер. 2002.-734с.
8. Дворогин А.И. Основы трансляции. Учебное пособие. — Волгоград. ВолГТУЛ 999.80г.