ФАЙЛДАР

C++ тилинде файл катары логикалык байланышта болгон информациялардын атайын уюштурулган жыйындысын айтышат. Файлдардын мисалы катары: баштапкы модулдун тексттин, аткарууга даяр болгон б.а. курагычтан (компоновкадан) кийинки программаны жана программа үчүн баштапкы маалыматтардын тобун алсак болот. Файл катары эстин кандайдыр бир информация кармаган бөлүгүн да алышат б.а. файлга маалыматтарды киргизип чыгаруу иш аракети катары карашат. C++ тилинде киргизип-чыгаруу библиотекалык функциялар аркылуу жүргүзүлөт. Киргизүү-чыгарууда файл (file) жана агым (stream) деген эки түшүнүктү айырмалап кароо керек. C++ системасы киргизип-чыгаруунун кайсыл түзүлүшү пайдаланаарына көз каранды болбогон киргизип-чыгарууну да колдонот.

Ушундай абстрактуу киргизип-чыгаруунун логикалык түзүлүшү – ***агым*** деп аталат.

Маалыматтарды белгилүү болгон ыкмалар менен сактоого жөндөмдүү болгон, аныкталган физикалык түзүлүштүн (дискалар, лазердик түзүлүштөр ж.б.) бөлүгүн ***файл*** деп атайбыз.

Демек баардык агымдар бирдей болот, ал эми файлдар дайыма эле бирдей боло бербейт. Агым-анык бир физикалык түзүлүшкө көз каранды болбогондуктан, агымга маалыматтарды сактоочу бир эле стандарттык камтылган программа ошол эле маалыматтарды дискага да, экранга да ж.б. түзүлүштөргө да чыгара алат.

C++ тилинде эки түрдөгү тексттик (text) жана экилик код түрүндөгү (binary) файлдар каралат.

Агым түшүнүгү

Көпчүлүк программалар киргизилүүчү маалыматтарды агым деп аталган атайын файлдардан окутат. Ошондой эле чыгарылуучу маалыматтар да агымга жазылат.

Агым деп – колдонуучунун деңгээлинде ийкемдүү киргизип- чыгаруу ишке ашыруучу буфер менен биргелешкен файлды айтабыз.

Буфер – бул учурдагы эстин бир бөлүгү. Буфердик киргизүүдө маалымат адегенде буферге келип түшөт андан кийин аны колдоно турган программага берилет. Ушундай эле буфердик чыгарууда маалымат программадан буферге берилет, андан кийин гана сырткы сактоочу түзүлүштөргө жиберилет.

Агымдарды аныктоо.

Агымдарды башкаруу үчүн агым көрсөткүчү колдонулат (stream pointer). Агым көрсөткүчү маалыматтарды кармаган файл жайгаштырылган түзүлүшкө болгон көрсөткүч (же шилтеме) катары эсептелет. Агым көрсөткүчү алдын ала аныкталган FILE типтүү түзүлүштүн жардамы менен аныкталат.

Төмөндөгүдөй: FILE * <агым көрсөткүчүнүн ысымы>

Агым көрсөткүчүнүн ысымы-бул FILE түзүлүшүнө болгон көрсөткүчтүн ысымы. FILE тибинин баяндалышы stdio.h файлында берилет.

Стандарттык агымдар.

C++ системасында үч стандарттык агымдар бар: stdin-
стандарттык киргизүү файлы.
stdout-стандарттык чыгаруу файлы.
stderr-каталар жөнүндө кабарлардын стандарттык файлы.

Бул файлдар дайыма колдонуучулардын терминалдарын (дисплей жана клавиатура) байланыштырат. Стандарттык файлдар программада иштегенде автоматтык түрдө ачылып, ал эми иштөөсүн токтотконду жабылат.

Стандарттык агымдар stdio.h файлында жатат жана аларды аныктоо FILE *stdin, *stdout, *stderr; түрүндө болот. Маалыматтарды стандарттык агымдардан окуу үчүн төмөндөгү функцияларды пайдаланса болот: getchar, gets жана калыптап окуучу scanf, ал эми маалыматтарды жазуу үчүн putchar, puts жана printf функциялары колдонулат.

Стандарттык эмес агымдар.

Программист аныктаган агымдар стандарттык эмес агымдар деп аталат. Эгерде колдонуучу стандарттык эмес агымдарды колдонсо, анда ал өзүнүн программасында ал агымды ачуу жана жабуу мүмкүнчүлүгүн карашы керек. Агымдар FILE тибин колдонуу менен аныкталат. Мисалы: FILE *shilteme

Файлды агым катары ачуу менен бул файл үчүн буферди уюштурууда fopen функциясы колдонулат. Файлды башкача жол менен да уюштурууга болот. Адегенде open функциясынын жардамы менен ачып, андан кийин аны fopen функциясы менен агымга өзгөртүп түзсө болот.

Файлдын көрсөткүчү маалыматтарды киргизүүдө жана чыгарууда агымдан кийинки элементинин абалын белгилейт. Файлдын көрсөткүчүн керектүү орунга орнотуу үчүн fseek- функциясы колдонулат. Ал эми файлдын көрсөткүчүн файлдын башына жылдырыш үчүн rewind-функциясы колдонулат.

Агымды ачууда fopen-функциясы колдонулат, жалпы жазылышы FILE *fopen (char *fisim, char *type); Бул функция fisim сабы менен берилген ысым менен файлды ачат, ал эми type параметри файлды ачуудагы колдонуучу режимдерди кармайт. Функция ачылган файлга көрсөткүчтү пайда кылат. Бул көрсөткүч FILE түзүлүшүн көрсөткөн көрсөткүч катары алдын ала баяндалышы керек. Эгерде жок файлды ачуу иш аракети каралса же кандайдыр бир ката келип чыкса бул функция NULL деген маанини берет. Ошондуктан fopen-функциясына кайрылуудан кийин программада шартуу if-оператору менен NULL мааниси барбы же жокпу текшерүү зарыл.

Мисалы:

FILE *shilteme;

shilteme=fopen (“misal.txt”, “w”);

if shilteme==NULL) printf (“файл ачылган жок”) else printf (“файл ачылды”)

Бул мисалда учурдагы каталогко misal.txt файлы маалыматтарды жазуу үчүн (“w”-маалыматтарды жазуу үчүн файлды түзүү режими) түзүлөт.

Файлды уюштуруу

Файлды уюштуруу үчүн fopen-функциясы колдонулат.

Жалпы жазылышы:

file * fopen (char * fisym, char * type);

Бул функция fisym-ысымдуу файлды ачат, ал эми type параметри ачылган файл колдонгон режимди аныктоочу сапты көрсөтөт. Файлды уюштуруу катары маалыматтарды сактоо үчүн файлды түзүүнү, маалыматтарды окуу үчүн файлды ачууну, маалыматтарды толуктоо үчүн файлды ачуу, файлды өркүндөтүү ж.у.с. файлдарды уюштуруунун параметрлери.

“r” – файлды окуу үчүн ачуу

“w”- файлга жазуу үчүн аны түзүү

“a”- файлга маалыматтарды толуктоо

“r+”- файлды өркүндөтүү б.а. окуу жана жазуу үчүн файлды ачуу

“w+”- файлды түзүү менен өркүндөтүү “a+”- файлды толуктоо менен өркүндөтүү **“r”** – параметрин колдонгондо:

сырткы эсте бар болгон файл ачылат.

эгерде сырткы эсте файл жок болсо fopen функциясы NULL деген маанини берет

“w”- параметрин колдонгондо:

сырткы эсте файл бар болсо ал файлдагы маалыматтар толугу менен өчүрүлөт да жаңы маалыматтар башынан баштап жазылат.

эгерде сырткы эсте андай ысымдуу файл жок болсо анда жаңы файл түзүлөт.

“a”- параметрин колдонгондо сырткы эстеги бар болгон файлга маалыматтарды аягынан тартып толуктайт, эгер жаңы файл болсо башынан баштап маалыматтар жазылат.
Өркүндөтүү параметрлерин (r+, w+, a+) колдонгондо файлды ачуу, өркүндөтүлбөгөн параметрлердегидей (r, w, a) эле болот. Бирок, биринчиден киргизүүдөн кийин ошол эле замат чыгаруу fseek же rewind функцияларын колдонгондон кийин гана ишке ашып fseek-функциясы файлдын көрсөткүчүн орнотот. Rewind- функциясы файлдын башталышына көрсөткүчтү орнотот. Экинчиден fseek же rewind функцияларын колдонбой туруп чыгаруудан кийин ошол эле замат киргизүүнү ишке ашырууга болбойт.

Агымды жабуу жана тазалоо

Агымды иштеткенден кийин аны жабыш керек. Жабуу үчүн fclose функциясына кайрылуу жүргүзүлөт. Бул функциянын жазылышы int fclose (FILL * agym); бул функция agym ысымдуу агымды жабат. Агымды жабуу алдында, бул агым менен байланышкан баардык буферлер тазаланат. Агымды тазалоо fflush-функциясы аркылуу жүргүзүлөт. Бул функциянын баяндалышы stdio.h файлында жайгашкан жана жалпы берилиши:
int fflush (FILE * agym)
Эгерде agym киргизүүнүн ачылган агымы болсо, agym ачылган бойдон калат. Ал эми чыгаруунун ачылган агымы болсо, бул агым менен байланышкан буферлер тазаланат.
Мисалы программанын бир бөлүгүн карайлы. scanf (“%d”, &a);
fflush (stdin); scanf (“%d”,
&b); fflush (stdin); gets
(sap);
Бул программанын бөлүгүндө биринчи колдонулган fflush (stdin) функциясы киргизүү туура эмес болгондо stdin-киргизүү агымын тазалайт. Экинчи scanf-функциясын колдонгон учурда да stdin-агымы тазаланат, себеби gets-функциясы так аткарылышы үчүн анын алдында fflush функциясын колдонуу керек.

Агымга маалыматтарды калыптап жазуу.

Маалыматтарды агымга жазуу үчүн fprintf-функциясы колдонулат. Жалпы жазылышы:
int fprintf (FILE * agym, char * format[, belgi, …,]);
fprintf-функциясы берилген чыгаруу агымына маалыматтарды жазат. Агымдын ысымы agym параметри менен берилет.
Мисалы:
FILE *kor;
kor=fopen (“mat1.mal”, “w”);
fprintf (kor, “%d %d”, n,m);
fprintf (kor, “%d”, x);} fclose (kor);

Агымдан маалыматтарды калыптап окуу үчүн fscanf-функциясы колдонулат. Жалпы жазылышы:
int fscanf (FILE * agym, char * format[, belgi, …,]);
fscanf-функциясы берилген кирүү агымынан маалыматтарды окуйт.
Мисалы:
FILE *kor;
kor=fopen (“mat1.mal”, “r”);
fscanf (kor, “%d %d”, &n, &m); for (i=0; i<n;
i++)
for (j=0; j<m; j++)
fscanf (kor, “%d”, &mat [i] [j]); fclose (kor);

Агымга маалыматтарды калыптап киргизүүгө жана чыгарууга мисал карайлы. Мейли

баштапкы матрица (mxn) өлчөмүндө берилсин жана бул матрица mat1.mal ысымдуу файлына жазылсын, ар бир сабынын элементтеринин суммасын эсептеп, ал суммалардын өсүү тартиби боюнча саптар ирээтелип алынган жаңы матрица mat2.mal файлына жазылсын. Бул мисалдын программасы төмөндөгүдөй.

```
# include <stdio.h> # include
<conio.h> main()
{
FILE *kor;/*баштапкы матрицаны кармаган mat1.mal файлын түзүүчү*/
        int n,m,i, j, x; clrscr();
        kor=fopen("d:mat1.mal","w");          /* d дискетине mat1.mal
файлын түзүү*/
        printf ("\n n<10, m<10 болгондой mat[n][m] матрицанын
        өлчөмдөрүн үтүр менен ажыратып киргиз");
        scanf ("%d,%d", &n,&m);
        printf("Бүтүн маанилүү өлчөмдөрү (%d*%d) болгон
        матрицаны сап боюнча киргиз:\n", n,m);
        for (i=0; i<n; i++) for
        (j=0; j<m; j++)
        {scanf ("%d", &x);
                fprintf (kor, "%2d", x);}
                fclose (kor);
        }
```

Жыйынтык:

n<10, m<10 болгондой mat[n] [m] матрицанын өлчөмдөрүн киргиз: 4,3
бүтүн маанилүү өлчөмдөрү (4*3) болгон матрицаны сап боюнча киргиз:

```
7       8   9
4       5   6
1       2   3
9       9   9
```

D: дискетинде mat1.mal файлы түзүлөт. Ал файлды тексттик редакторлор менен ачып көрсөк болот mat1.mal файлы кармаган маалыматтар төмөндөгүдөй түрдө болот.