ТУУНДУ ТИПТЕР

Массивтер

Маалыматтардын туунду типтери деп, негизги типтерден маалыматтардын куралып түзүлүшүн айтабыз. Программа түзүүдө бир типтеги көп өзгөрмөлөр менен иштөөгө туура келет. Көп өзгөрмөлөрдүн ысымдарын программага колдонуу, анын көлөмүн чоңойтуп иштөөсүнүн тездигин азайтат, ушул себептен улам массив түшүнүгү программа түзүүдө колдонулат.

Массив - деп бир ысымдуу, бир типтеги, чектелген, ирээттелген жана өлчөмгө ээ болгон маалыматтардын жыйындысын айтабыз.

Массив түшүнүгү беш мүнөздөмөгө ээ болот: массивтин ысымы, элементтеринин номери, элементтеринин типтери, өлчөмү жана элементтеринин саны.

Массивтин элементтеринин номерин индекс деп да аташат.

Си тилинде массив төмөндөгүдөй жарыяланат:

<тип><массивтин ысымы.> [n1][n2]…[nk]

Бул жазууда <тип>-массивтин элементтеринин тиби. n1,n2,…nk – массивтин өлчөмдөрү. Массив бир өлчөмдүү болсо a[n]; эки өлчөмдүү болсо матрица деп аталып b[n] [m], массив үч өлчөмдүү болсо c[n] [m] [k] деп жазылат. Массивтердин индекстери 0 дөн башталат.Мисалы int a[15] ; берилсе анда:

a[0] - 1 элементи, a[1] -
2 элементи, a[2] - 3
элементи

….

a[13]- 14 элементи

a[14]- 15 элементи деп эсепетелинет.

Ал эми эки өлчөмдүү массивти же матрицаны карасак.

Мисалы float b[2] [3] .

Бул матрица эки сап жана үч мамычадан турат. Элементтери калкыма чекиттүү болуп, учурдагы эсте төмөндөгүдөй тартип менен жайгаштырылат b[0][0]; b[0][1]; b[0][2]; b[1][0]; b[1][1]; b[1][2].

Си тилинде массив үчүн учурдагы эстен дайыма орун бөлүнөт жана массивтин индекстери чектен чыгып кетүүсү текшерилбейт. Мисалы: int a[50] деп жарыяланса, ал эми программада a[100] көрсөтүлсө, бул катага алып келбейт. Себеби a[100] элементи катары массивтин башынан баштап кайрадан менчиктелүү жүрөт. Индекс жок массивтин ысымы массивтин башталышынын адресин берген туруктуу (константа) болуп эсептелет.

Эки өлчөмдүү массивтин учурдагы эсте жайгашаарын карайлы.

Мисалы: int a[2][3] анда жогорудагыдай а массивинин элементтери төмөндөгүдөй жайгаштырылат:

a[0][0], a[0][1], a[0][2], a[1][0], a[1][1], a[1][2] эгерде a=1000

болсо, анда a[0][1] элементинин адреси 1002 болот, себеби int тиби эсте 2 байт өлчөмдү ээлейт, ал эми массивтин кийинки элементи a[0][2] болсо1004 адреске ээ болот ж.у.с.

Массивтер менен иштөөдө ага маанилештирүү иш- аракеттерин да жүргүзүүгө болот. ***Маанилештир*** - деп массивти баяндоодо анын маанилерин б.а. элементтерин кошо берүүнү айтабыз.

Мисалы: int b[3]={1,2,3} бул учурда b[0]= 1; b[1]=2; b[2]=3 маанилерин алышат. Көп өлчөмдүү массивтерди маанилештирүүгө *мисал:*

int a[2][3]={1,2,3,4,5,6} бул маанилештирүү төмөндөгү менчиктөө операторлорунун тобуна тең күчтүү болуп калат.

a[0][0]=1; a[0][1]=2; a[0][2]=3; a[1][0]=4; a[1][1]=5; a[1][2]=6;

Көпчүлүк учурда көп өлчөмдүү массивтерге маанилештирүүдө кабатталган маанилештирүүнү да колдонушат.

Мисалы: int a[2][3]={{1,2,3},{4,5,6}}.

Кээ бир учурда программада окулушу жөнөкөй болуш үчүн төмөндөгүдөй да маанилештирүүнү колдонушат.

```
int mas[5][3]={{1,1,1},
               {2,2,2},
               {3,3,3},
               {4,4,4},
               {5,5,5}};
```

Символдук массивтерди да маанилештирсе болот.

Мисалы: char s[10]={‘t’,’u’,’r’,’b’,’o’,’ ‘,’c’}

Сап маанилүү өзгөрмөлөрдү да маанилештирүүнү карайлы.

Мисалы: char sap1[7]=”Салам!”

Саптын узундугу 7, себеби саптын аягын билдирүүчү символго бир орун берилет. Саптын узундугун чарчы кашаада жазбай койсо да болот.

Мисалы: char sap2[ ]={‘c’,’a’,’л’,’a’,’м’,’!’,’\o’};

Жогорудагы сап массивин маанилештирүүнүн стандарттык формасы болуп эсептелет. Бул формадагы ‘\o’ белгисинин бар болушу сап массиви экендигин далилдеп турат, эгер бул белги жок болсо анда белгилердин гана массиви болуп калмак.

Массивтерди түзүүгө жана анын элементтерин иштетүү карата мисалдарды карайлы. Мейли бир өлчөмдүү массивтин эң чоң элементтин жана терс элементтеринин санын аныктоочу программаны түзөлү.

```
#include <stdio.h> #include
<conio.h >
main( )
{int a[15],          /*a масивин баяндоо*/
n,                   /*a массивин элементтерин санын аныктоо*/
i,                   /*массивтин индекси*/
max,                 /*a массивинин элементтеринин максималдуу мааниси*/
k;                   /*a массивинин терс элементтеринин саны*/
/*a массивинин элементтерин киргизүү*/ clrscr();
do
{
printf(“15 тен ашпаган элементтердин санын бер n=”); scanf(“%d”,&n);
}
while(n<1|| n>15);
for(i=0; i<n; i++){ printf(“Масситин %d-элементин киргиз”,i+1);
                    scanf(“%d”,&a[i]);};
/*a массивинин эң чоң элементин аныктоо*/ max=a[0];
for(i=1;i<n;i++);
if(max<a[i])max=a[i];
printf(“\n Массивтин эң чоң элементи=%d” , max);
/*a массивинин терс элементтеринин санын аныктоо*/ k=0;
```

```
for (i=0;i<n ;i++) if (a[i]<0)k++;
printf (“\n Массивтин терс элементтеринин саны=%d”,k);
}
```

Жыйынтык:

15 тен ашпаган элементтердин санын бер n=5

Масситин 1-элементин киргиз 2 Масситин 2-элементин киргиз 25 Масситин 3-элементин киргиз -8 Масситин 4-элементин киргиз 7

Масситин 5-элементин киргиз -4 Массивтин эң чоң элементи=25 Массивтин терс элементтеринин саны=2

Массивтерди маанилештирүүгө мисал. Мейли a жана b матрицалары (эки өлчөмдүү массивтер) берилсин бул матрицалардын суммасын тапкыла:

```
#include <stdio.h>
#include <conio.h >
main( )
{int  a[3][4]={{1,1,1,1},{2,2,2,2},{3,3,3,3}},
b[3][4]={{4,4,4,4},{5,5,5,5},{6,6,6,6}},    c[3][4],i,j;
clrscr(); for(i=0;i<3;i++)
for(j=0;j<4;j++)  c[i][j]=a[i][j]+b[i][j];
/*C матрицасын чыгаруу */ for (i=0;
i<3; i++);
{for (j=0; j<4; j++);              printf(“c[%d] [%d]=%d”, i, j,c[i][j]); printf(“\n”);
};
}
```

Жыйынтык:

```
c[0][0]=5
c[0][1]=5
c[0][2]=5
c[0][3]=5
c[1][0]=7
c[1][1]=7
c[1][2]=7
c[1][3]=7
c[2][0]=9
c[2][1]=9
c[2][2]=9
c[2][3]=9
```

Көрсөткүчтөр менен массивтердин ортосундагы байланыш

C++ тилинде массивтер менен көрсөткүчтөр тыгыз байланышта. Массивтин ысымын массивтин башталыш элементинин адресин көрсөтүүчү катары караса болот. Тактап айтканда массивтин ысымы индекси жок жазылса мисалы a – бир өзгөрмөлүү массив болсо “a” жазуусу массивтин нөлүнчү элементинин адресин көрсөтөт б.а. “& a[0]” жазуусун билдирет. Ал эми көрсөткүчтөрдүн арифметикасы боюнча “a+i” –жазуусу i-

элементтин адресин билдирет, ошондой эле “*(a+i)” – жазуусу “a[i]” жазуусуна тең күчтүү. Жалпылаганда төмөндөгү тең күчтүүлүк орун алат:

```
a==&a[0] (a+i
)==&a[i]
*(a+i)==a[i]
```

Бул жазуулар солдон оңду жана оңдон солду көздөй колдонуу туура болуп эсептелет. Массивтин ысымынын же көрсөткүч же массив катары баяндалышынын негизги айырмасы, массив өзүнүн жайланышына карата болот. Эгерде өзгөрмө массив катары жарыяланса, анда программа автоматтык түрдө керектүү өлчөмдө эстен орун бөлөт, ал эми массивтин ысымы көрсөткүч катары жарыяланса, анда программа calloc функциясын колдонуу менен эстен ушул массив үчүн орун бөлөт же мурун аныкталган эстин сегментинен массивтин ысымы үчүн адрес менчиктейт.

Өзгөрмөнүн массив катары баяндалышына мисал карайлы:

```
#include <stdio.h>
#include <conio.h > main ( )
{int  a[3],i; clrscr();
a[0]=30;
a[1]=7; a[2]=2003;
printf(“адрестин тизмелери:”); for(i=0;i<3;i++)
printf(“%4p  ”,&a[i]); printf(“\n маанилердин
тизмеси:”); for(i=0; i<3; i++) printf (“%-4d”,a[i]);
}
```

Жыйынтык:

```
адрестин тизмелери: FFF0  FFF2  FFF4
маанилердин тизмеси:          30      7        2003
```

Кийинки *мисалда* а массиви көрсөткүч катары баяндалышын карайлы:

```
#include        <stdio.h>
#include        <alloc.h>
#include <conio.h >
main ( )
{int *pa,i;
clrscr();
pa=(int*)calloc(3,sizeof(int));
*pa=30;
*(pa+1)=7;
*(pa+2)=2003;
printf( “адрестердин тизмеси:”);
for(i=0;i<3;i++) printf(“%4p ”,(pa+i)); printf(“\n
маанилердин тизмеси:”);
for (i=0;i<3;i++) printf(“%-4d ”, *(pa+i));
}
```

Жыйынтык:

```
адрестин тизмелери: 05D0  05D2  05D4
маанилердин тизмеси: 30          7        2003
```

Жогорудагы эки *мисалды* бир программа катары кароого болот:

```
#include <stdio.h>
#include <alloc.h>
main ( )
 {int *pa, a[3],i; pa= a;
   *pa=30;
   *(pa+1)=7;
   *(pa+2)=2003;
   printf("адрестердеги маанилердин тизмеси:"); for(i=0;i<3;i++) printf("%-
   4d",*(pa+i));
}
```

Жыйынтык:

адрестердеги маанилердин тизмеси: 30 7 2003

Бул мисалда pa=a; операторуpa көрсөткүчүнө a статикалык массивинин башталышынын адресин менчиктейт, ал эми a=pa; жазылышы туура эмес жана каталыка алып келет, себеби бул учурда a статикалык массивинин адресин өзгөртүү аракети жүргүзүлүп калат.

Массивтин ысымы дайыма адреске болгон шилтеме катары каралат. Эгерде мындай шилтеменин көчүрмөсүн көрсөткүчкө

ыйгарсак, көрсөткүчтүн маанисин чоңойтуу же кичирейтүү менен массивтин кандайдыр бир элементине шилтеме жүргүзүүгө болот

Мисалы:

```
int  b[5], *pb, i;
pb=b;   /*pb  көрсөткүчү  b[0]  дү  көрсөтүп  турат*/  pb++;   /*pb
көрсөткүчү b[1] ди көрсөтүп турат*/ pb++;      /*pb  көрсөткүчү  b[2]
ни көрсөтүп турат*/
for (i=0;i<5; i++) *pb++=0  /*b  массивинин элементинин баарысы 0 болот*/
```

Жогорку программанын фрагментиндеги акыркы сабындагы

*pb++=0 операторунун аткарылышын карайлы. Адегенде * - адреске кайрылуу амалы pb адреси боюнча маанини берет. Бул маани 0 гө барабар болот. Циклдин биринчи айлампасында b[0] элементи нөлгө айланат, андан ары pb адреси "++" амалы аркылуу бирге чоңоёт.

Массивтер жана көрсөткүчтөр менен иштөөдө олуттуу каталар болуп маанилештирилбеген көрсөткүчтү колдонуу жана массивтин чегинен чыгып кетүү эсептелинет int *p көрсөткүчтү баяндаса аны маанилештирүү деп программада

p=(int *) malloc(sizeof(int)); сабынын жазылышын түшүнөбүз. Эки өлчөмдүү массивтер жана көрсөткүчтөр.

Эки өлчөмдүү массив бул ар бир элементи сап болгон бир өлчөмдүү массивти түшүнөбүз.

Мисалы: a[3] [4] болсо:

a – массивтин нөлүнчү (0) сабынын көрсөткүчү. (a+1) - массивтин 1- сабынын көрсөткүчү.

(a+2) - массивтин 2- сабынын көрсөткүчү ж. у. с.

a[2] [3] жазылышы *(a[2]+3) же *(*(a+2)+3) жазылыштары менен эквиваленттүү.

Жалпылаганда a[i] [j] матрицасы компилятор тарабынан

((a+i)+j) – эквиваленттүү жазылышына которулат.