

CS200 – Програмчлалын үндэс

Лекц 05

Массив

Профессор А.Эрдэнэбаатар

Лекцийн агуулга

- ▶ Массивыг тодорхойлох, идэвхижүүлэх, элементэд хандах
- ▶ Тэмдэгт тогтмолыг тодорхойлох
- ▶ Массивыг функцэд дамжуулах
- ▶ Жагсаалт, хүснэгтийн хадгалалт, эрэмбэлэлт, хайлтанд массивыг ашиглах
- ▶ Олон хэмжээст массивыг тодорхойлох, ашиглах
- ▶ Дүгнэлт

Массив

► Массив

- Санах ойн дараалсан бүлэг байршил
- Нэг нэр, төрөл

► Массивын элементэд хандахдаа

- Массивын нэр
- Байршлын дугаар

► Загвар

arrayname[position number]

- **n** элементтэй **c** массив:
c[0], c[1]...c[n-1]

Массивын нэр(элемент бүр адилхан нэртэйг анхаар)



c[0]	-41
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78



Массив дахь байршлын дугаар

Массив

- ▶ Массивын элемент энгийн хувьсагчтай адилхан

```
c[0] = 3;
```

```
Printf( "%d", c[0] );
```

- ▶ Индекс дээр үйлдэл хийж болно. Хэрэв $x=3$ бол

```
c[5-2], c[3], c[x] – бүгд адилхан
```

- ▶ Анхаарах зүйлс:

- ▶ “Массивын 7 дахь элемент” гэдэг массивын 6 гэсэн индекстэй элемент
- ▶ “Массивын элемент 7” гэдэг массивын 7 индекстэй буюу 8 дахь элемент гэсэн үг

Үйлдлүүдийн ахлах чанар

Үйлдэл	Биелэгдэх чиглэл	Төрөл
[] ()	Зүүнээс баруун	Хамгийн ахлах
+ - ! (төрөл) ++ --	Баруунаас зүүн	Ганц гишүүнт
* / %	Зүүнээс баруун	Үржигдэх
+ -	Зүүнээс баруун	Нэмэгдэх
< <= > >=	Зүүнээс баруун	Харьцаа
== !=	Зүүнээс баруун	Тэнцэтгэл
&&	Зүүнээс баруун	Логик AND
	Зүүнээс баруун	Логик OR
?:	Баруунаас зүүн	Нөхцөлт
= += -= *= /= %=	Баруунаас зүүн	Олгох
,	Зүүнээс баруун	Таслал

Массивыг тодорхойлох

- ▶ Массивыг тодорхойлохдоо

- ▶ Нэр

- ▶ Төрөл

- ▶ Элементийн тоо

- `arrayType arrayName[numberOfElements] ;`

- ▶ Жишээ:

- `int c[10] ;`

- `float myArray[3284] ;`

- ▶ Нэг төрлийн олон массивыг тодорхойлох

- ▶ Загвар нь энгийн хувьсагчтай төстэй

- ▶ Жишээ:

- `int b[100] , x[27] ;`

Массивын жишээ

- ▶ Идэвхижүүлэлт

```
int n[10] = { 1, 2, 3, 4, 5 };
```

- ▶ Идэвхүүлэлтийн тоо цөөн бол үлдсэн нь 0 болдог

```
int n[5] = { 1 } гэвэл эхний элементээс бусад нь 0
```

- ▶ Идэвхүүлэлтийн тоо хэт их бол дүрмийн алдаа гарна
- ▶ Си хэлэнд массивын хязгаарыг шалгадаггүй

- ▶ Хэрвээ хэмжээг орхигдуулсан бол идэвхижүүлэлт түүнийг тодорхойлдог

```
int n[ ] = { 1, 2, 3, 4, 5 } гэвэл массив 5  
элементтэй гэсэн үг
```

Массивын идэвхижүүлэлт

```
1.  /* Ex_25 initializing an array */
2.  #include <stdio.h>
3.
4.  /* function main begins program execution */
5.  int main( void )
6.  {
7.      int n[ 10 ]; /* n is an array of 10 integers */
8.      int i; /* counter */
9.
10.     /* initialize elements of array n to 0 */
11.     for ( i = 0; i < 10; i++ ) {
12.         n[ i ] = 0; /* set element at location i to 0 */
13.     } /* end for */
14.
15.     printf( "%s%13s\n", "Element", "Value" );
16.
17.     /* output contents of array n in tabular format */
18.     for ( i = 0; i < 10; i++ ) {
19.         printf( "%7d%13d\n", i, n[ i ] );
20.     } /* end for */
21.
22.     return 0; /* indicates successful termination */
23.
24. } /* end main */
```

for цикл массивын элементүүдийг
ганцаарчилан идэвхижүүлж байна

for циклээр массивын бүх
элементүүдийг гаргаж байна

Массивын идэвхижүүлэлт...

Element	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

Массивын идэвхижүүлэлт

```
1.  /* Ex_26 Initializing an array with a initializer list */
2.  #include <stdio.h>
3.
4.  /* function main begins program execution */
5.  int main( void )
6.  {
7.      /* use initializer list to initialize array n */
8.      int n[ 10 ] = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };
9.      int i; /* counter */
10.
11.     printf( "%s%13s\n", "Element", "Value" );
12.
13.     /* output contents of array in tabular format */
14.     for ( i = 0; i < 10; i++ ) {
15.         printf( "%7d%13d\n", i, n[ i ] );
16.     } /* end for */
17.
18.     return 0; /* indicates successful termination */
19.
20. } /* end main */
```

Жагсаалтаар массивын бүх элементүүдийг нэг зэрэг идэвхижүүлж байна

Массивын идэвхижүүлэлт...

Element	Value
0	32
1	27
2	64
3	18
4	95
5	14
6	90
7	70
8	60
9	37

Лекцийн агуулга

- ▶ Массивыг тодорхойлох, идэвхижүүлэх, элементэд хандах
- ▶ Тэмдэгт тогтмолыг тодорхойлох
- ▶ Массивыг функцэд дамжуулах
- ▶ Жагсаалт, хүснэгтийн хадгалалт, эрэмбэлэлт, хайлтанд массивыг ашиглах
- ▶ Олон хэмжээст массивыг тодорхойлох, ашиглах
- ▶ Дүгнэлт

Тэмдэгт тогтмолыг тодорхойлох

```
1.  /* Initialize the elements of array s to the even integers from 2 to 20 */
2.  #include <stdio.h>
3.  #define SIZE 10
4.
5.  /* function main begins program execution */
6.  int main( void )
7.  {
8.      /* symbolic constant SIZE can be used to specify array size */
9.      int s[ SIZE ]; /* array s has 10 elements */
10.     int j; /* counter */
11.
12.     for ( j = 0; j < SIZE; j++ ) { /* set the values */
13.         s[ j ] = 2 + 2 * j;
14.     } /* end for */
15.
16.     printf( "%s%13s\n", "Element", "Value" );
17.
18.     /* output contents of array s in tabular format */
19.     for ( j = 0; j < SIZE; j++ ) {
20.         printf( "%7d%13d\n", j, s[ j ] );
21.     } /* end for */
22.
23.     return 0; /* indicates successful termination */
24.
25. } /* end main */
```

#define удирдамж хөрвүүлэгч SIZE гэдэг үг гарах бүрт түүнийг 10 –р соль гэдгийг заана

SIZE 10-р солигдох тул массив 10 элементтэй

for цикл массивын элементүүдийг ганцаарчилан идэвхижүүлж байна

Тэмдэгт тогтмолыг тодорхойлох...

Element	Value
0	2
1	4
2	6
3	8
4	10
5	12
6	14
7	16
8	18
9	20

Тэмдэгт тогтмолыг тодорхойлох

```
1.  /* Ex_28 Compute the sum of the elements of the array */
2.  #include <stdio.h>
3.  #define SIZE 12
4.
5.  /* function main begins program execution */
6.  int main( void )
7.  {
8.      /* use initializer list to initialize array */
9.      int a[ SIZE ] = { 1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45 };
10.     int i; /* counter */
11.     int total = 0; /* sum of array */
12.
13.     /* sum contents of array a */
14.     for ( i = 0; i < SIZE; i++ ) {
15.         total += a[ i ];
16.     } /* end for */
17.
18.     printf( "Total of array element values is %d\n", total );
19.
20.     return 0; /* indicates successful termination */
21.
22. } /* end main */
Total of array element values is 383
```

Жагсаалтаар массивын бүх элементүүдийг нэг зэрэг идэвхижүүлж байна

for цикл массивын элемент бүрийг total хувьсагч дээр нэмж байна

Тэмдэгт тогтмолыг тодорхойлох

```
1.  /* Ex_29 Student poll program */
2.  #include <stdio.h>
3.  #define RESPONSE_SIZE 40 /* define array sizes */
4.  #define FREQUENCY_SIZE 11
5.
6.  /* function main begins program execution */
7.  int main( void )
8.  {
9.      int answer; /* counter to loop through 40 responses */
10.     int rating; /* counter to loop through frequencies 1-10 */
11.
12.     /* initialize frequency counters to 0 */
13.     int frequency[ FREQUENCY_SIZE ] = { 0 };
14.
15.     /* place the survey responses in the responses array */
16.     int responses[ RESPONSE_SIZE ] = { 1, 2, 6, 4, 8, 5, 9, 7, 8, 10,
17.                                         1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7, 5, 6, 6,
18.                                         5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };
19.
20.     /* for each answer, select value of an element of array responses
21.        and use that value as subscript in array frequency to
22.        determine element to increment */
23.     for ( answer = 0; answer < RESPONSE_SIZE; answer++ ) {
24.         ++frequency[ responses [ answer ] ];
25.     } /* end for */
```

#define удирдамж тэмдэгт тогтмолыг тодорхойлдог

frequency массив 11 элементтэй гэж тодорхойлогдлоо

40 элементтэй **responses** массивын элементүүд идэвхижиж байна

frequency массивын индексийг **responses** массивын утга тодорхойлж байна

Тэмдэгт тогтмолыг тодорхойлох...

```
26.  
27.     /* display results */  
28.     printf( "%s%17s\n", "Rating", "Frequency" );  
29.  
30.     /* output the frequencies in a tabular format */  
31.     for ( rating = 1; rating < FREQUENCY_SIZE; rating++ ) {  
32.         printf( "%6d%17d\n", rating, frequency[ rating ] );  
33.     } /* end for */  
34.  
35.     return 0; /* indicates successful termination */  
36.  
37. } /* end main */
```

Rating	Frequency
1	2
2	2
3	2
4	2
5	5
6	11
7	5
8	7
9	1
10	3

ТЭМДЭГТ ТОГТМОЛЫГ ТОДОРХОЙЛОХ

```
1.  /* Ex_30 Histogram printing program */
2.  #include <stdio.h>
3.  #define SIZE 10
4.
5.  /* function main begins program execution */
6.  int main( void )
7.  {
8.      /* use initializer list to initialize array n */
9.      int n[ SIZE ] = { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 };
10.     int i; /* outer for counter for array elements */
11.     int j; /* inner for counter counts *s in each histogram bar */
12.
13.     printf( "%s%13s%17s\n", "Element", "Value", "Histogram" );
14.
15.     /* for each element of array n, output a bar of the histogram */
16.     for ( i = 0; i < SIZE; i++ ) {
17.         printf( "%7d%13d", i, n[ i ] );
18.
19.         for ( j = 1; j <= n[ i ]; j++ ) { /* print one bar */
20.             printf( "%c", '*' );
21.         } /* end inner for */
22.
23.         printf( "\n" ); /* end a histogram bar */
24.     } /* end outer for */
25.
26.     return 0; /* indicates successful termination */
27.
28. } /* end main */
```

Үүрлэсэн for цикл нь $n[i]$ тооны
одыг i дэх мөрөнд хэвлэж байна

Тэмдэгт тогтмолыг тодорхойлох...

Element	Value	Histogram
0	19	*****
1	3	***
2	15	*****
3	7	*****
4	11	*****
5	9	*****
6	13	*****
7	5	*****
8	17	*****
9	1	*

Тэмдэгт тогтмолыг тодорхойлох

```
1.  /* Ex_31 Roll a six-sided die 6000 times */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.  #include <time.h>
5.  #define SIZE 7
6.
7.  /* function main begins program execution */
8.  int main( void )
9.  {
10.     int face; /* random die value 1 - 6 */
11.     int roll; /* roll counter */
12.     int frequency[ SIZE ] = { 0 }; /* clear counts */
13.
14.     srand( time( NULL ) ); /* seed random-number generator */
15.
16.     /* roll die 6000 times */
17.     for ( roll = 1; roll <= 6000; roll++ ) {
18.         face = 1 + rand() % 6;
19.         ++frequency[ face ]; /* replaces 26-line switch of Fig. 5.8 */
20.     } /* end for */
```

for цикл шооны нүд бүрт зориулсан 6 хувьсагч болон switch оператор ашиглахын оронд нэг массив ашиглаж байна

Тэмдэгт тогтмолыг тодорхойлох...

```
21.  
22.     printf( "%s%17s\n", "Face", "Frequency" );  
23.  
24.     /* output frequency elements 1-6 in tabular format */  
25.     for ( face = 1; face < SIZE; face++ ) {  
26.         printf( "%4d%17d\n", face, frequency[ face ] );  
27.     } /* end for */  
28.  
29.     return 0; /* indicates successful termination */  
30.  
31. } /* end main */
```

Face	Frequency
1	1029
2	951
3	987
4	1033
5	1010
6	990

Массивын жишээ

▶ Тэмдэгт массив

- ▶ “first” мөр бол үнэндээ тэмдэгтийн статик массив юм
- ▶ Тэмдэгт массивыг тэмдэгт мөрөөр идэвхижүүлж болно

```
char string1[] = "first";
```

- ▶ Мөр Null буюу ‘\0’ –тэмтэгтээр төгсдөг. Ингэхээр мөр нь үнэндээ 6 элементтэй . Дараах мөр дээрхтэй адил

```
char string1[] = {'f','i','r','s','t'};
```

- ▶ Тухайлсан тэмдэгтэд хандаж болно

```
string1[3] гэдэг нь ‘s’ гэсэн үг
```

- ▶ Массивын нэр өөрөө массивын хаяг болдог учир `scanf` –т & хэрэггүй

```
scanf("%s", string2);
```

- ▶ Тусгай тэмдэгт (whitespace) гартал уншдаг, массивын төгсгөлийг давж магадгүй!

Тэмдэгт массивын жишээ

```
1.  /* Ex_32 Treating character arrays as strings */
2.  #include <stdio.h>
3.
4.  /* function main begins program execution */
5.  int main( void )
6.  {
7.      char string1[ 20 ]; /* reserves 20 characters */
8.      char string2[] = "string literal"; /* reserves 15 characters */
9.      int i; /* counter */
10.
11.     /* read string from user into array string1 */
12.     printf("Enter a string: ");
13.     scanf( "%s", string1 ); /* input ended by whitespace character */
14.
15.     /* output strings */
16.     printf( "string1 is: %s\nstring2 is: %s\n"
17.            "string1 with spaces between characters is:\n",
18.            string1, string2 );
19.
20.     /* output characters until null character is reached */
21.     for ( i = 0; string1[ i ] != '\0'; i++ ) {
22.         printf( "%c " string1[ i ] );
23.     } /* end for */
```

string2 массив бол элемент бүр нь нэг тэмдэгттэй(ө.х. \0 –г оролцуулаад 15 элемент)

for цикл **string1** массивын тэмтэгтүүдийг хооронд нь сул зай орхин хэвлэж байна

Тэмдэгт массивын жишээ...

```
24.  
25.     printf( "\n" );  
26.  
27.     return 0; /* indicates successful termination */  
28.  
29. } /* end main */
```

Enter a string: Hello there

String1 is: Hello

String2 is: string literal

String1 with spaces between characters is:

H e l l o

- ▶ Үйлчлэх мужаасаа олон дахин гарч, ордог функц доторх автоматик массивыг **static** гэж тодорхойлох/зарлах нь зохимжтой. Ингэснээр функцийг дуудах бүрт массивыг байгуулаад байх шаардлагагүй болно.

static МАССИВЫН ЖИШЭЭ

```
1.  /* Ex_33 Static arrays are initialized to zero */
2.  #include <stdio.h>
3.
4.  void staticArrayInit( void );    /* function prototype */
5.  void automaticArrayInit( void ); /* function prototype */
6.
7.  /* function main begins program execution */
8.  int main( void )
9.  {
10.     printf( "First call to each function:\n" );
11.     staticArrayInit();
12.     automaticArrayInit();
13.
14.     printf( "\n\nSecond call to each function:\n" );
15.     staticArrayInit();
16.     automaticArrayInit();
17.
18.     return 0; /* indicates successful termination */
19.
20. } /* end main */
21.
```

static МАССИВЫН ЖИШЭЭ...

```
22. /* function to demonstrate a static local array */
23. void staticArrayInit( void )
24. {
25.     /* initializes elements to 0 first time function is called */
26.     static int array1[ 3 ];
27.     int i; /* counter */
28.
29.     printf( "\nValues on entering staticArrayInit:\n" );
30.
31.     /* output contents of array1 */
32.     for ( i = 0; i <= 2; i++ ) {
33.         printf( "array1[ %d ] = %d  ", i, array1[ i ] );
34.     } /* end for */
35.
36.     printf( "\nValues on exiting staticArrayInit:\n" );
37.
38.     /* modify and output contents of array1 */
39.     for ( i = 0; i <= 2; i++ ) {
40.         printf( "array1[ %d ] = %d  ", i, array1[ i ] += 5 );
41.     } /* end for */
42.
43. } /* end function staticArrayInit */
44.
```

static МАССИВ НЭГ Л удаа
staticArrayInit-г дуудах үед үүснэ

static МАССИВЫН ЖИШЭЭ...

```
45. /* function to demonstrate an automatic local array */
46. void automaticArrayInit( void )
47. {
48.     /* initializes elements each time function is called */
49.     int array2[ 3 ] = { 1, 2, 3 };
50.     int i; /* counter */
51.
52.     printf( "\n\nValues on entering automaticArrayInit:\n" );
53.
54.     /* output contents of array2 */
55.     for ( i = 0; i <= 2; i++ ) {
56.         printf("array2[ %d ] = %d  ", i, array2[ i ] );
57.     } /* end for */
58.
59.     printf( "\nValues on exiting automaticArrayInit:\n" );
60.
61.     /* modify and output contents of array2 */
62.     for ( i = 0; i <= 2; i++ ) {
63.         printf( "array2[ %d ] = %d  ", i, array2[ i ] += 5 );
64.     } /* end for */
65.
66.     printf("\n");
67.
68. } /* end function automaticArrayInit */
```

АВТОМАТИК МАССИВ `automaticArrayInit`
—г дуудах бүрт дахин үүснэ

static МАССИВЫН ЖИШЭЭ...

First call to each function:

Values on entering staticArrayInit:

array1[0] = 0 array1[1] = 0 array1[2] = 0

Values on exiting staticArrayInit:

array1[0] = 5 array1[1] = 5 array1[2] = 5

Values on entering automaticArrayInit:

array2[0] = 1 array2[1] = 2 array2[2] = 3

Values on exiting automaticArrayInit:

array2[0] = 6 array2[1] = 7 array2[2] = 8

Second call to each function:

Values on entering staticArrayInit:

array1[0] = 5 array1[1] = 5 array1[2] = 5

Values on exiting staticArrayInit:

array1[0] = 10 array1[1] = 10 array1[2] = 10

Values on entering automaticArrayInit:

array2[0] = 1 array2[1] = 2 array2[2] = 3

Values on exiting automaticArrayInit:

array2[0] = 6 array2[1] = 7 array2[2] = 8

Лекцийн агуулга

- ▶ Массивыг тодорхойлох, идэвхижүүлэх, элементэд хандах
- ▶ Тэмдэгт тогтмолыг тодорхойлох
- ▶ Массивыг функцэд дамжуулах
- ▶ Жагсаалт, хүснэгтийн хадгалалт, эрэмбэлэлт, хайлтанд массивыг ашиглах
- ▶ Олон хэмжээст массивыг тодорхойлох, ашиглах
- ▶ Дүгнэлт

Массивыг функцэд дамжуулах

► Массивыг дамжуулах

- Массив аргументыг функцэд дамжуулахдаа түүний нэрийг ямар нэг хаалтгүй зааж өгдөг

```
int myArray[24];  
myFunction(myArray, 24);
```

- Функцэд массивын хэмжээг голдуу дамжуулдаг

- Массив хаягаар-дуудагдаж дамждаг
- Массивын нэр бол эхний элементийн хаяг юм
- Функц нь массив хаана хадгалагдаж байгааг мэддэг

► Массивын элементийг дамжуулах

- Элемент утгаар-дуудагдаж дамждаг
- Индекс тэй нэрийг (ө.х.: `myArray[3]`) дамжуулна

Массивыг функцэд дамжуулах

► Функцийн загвар

```
void modifyArray( int b[], int arraySize );
```

► Загварт параметрийн нэрийг заавал бичих албагүй

`int b[]` гэдгийг `int []` гэж бичиж болно

`int arraySize` гэдгийг `int` гээд хялбарчилж болно

Массивыг функцэд дамжуулах

```
1.  /* Ex_34 The name of an array is the same as &array[ 0 ] */
2.  #include <stdio.h>
3.
4.  /* function main begins program execution */
5.  int main( void )
6.  {
7.      char array[ 5 ]; /* define an array of size 5 */
8.
9.      printf( "      array = %p\n&array[0] = %p\n      &array = %p\n",
10.           array, &array[ 0 ], &array );
11.
12.      return 0; /* indicates successful termination */
13.
14. } /* end main */
    array = 0012FF78
    &array[0] = 0012FF78
    &array = 0012FF78
```

Санамж: %p гэдэг хувиргагч нь хаягийг 16-тын тоо байдлаар гарахыг зааж байна

Массивыг функцэд дамжуулах

```
1.  /* Ex_35 Passing arrays and individual array elements to functions */
2.  #include <stdio.h>
3.  #define SIZE 5
4.
5.  /* function prototypes */
6.  void modifyArray( int b[], int size );
7.  void modifyElement( int e );
8.
9.  /* function main begins program execution */
10. int main( void )
11. {
12.     int a[ SIZE ] = { 0, 1, 2, 3, 4 }; /* initialize a */
13.     int i; /* counter */
14.
15.     printf( "Effects of passing entire array by reference:\n\nThe "
16.            "values of the original array are:\n" );
17.
18.     /* output original array */
19.     for ( i = 0; i < SIZE; i++ ) {
20.         printf( "%3d", a[ i ] );
21.     } /* end for */
22.
23.     printf( "\n" );
24.
25.     /* pass array a to modifyArray by reference */
26.     modifyArray( a, SIZE );
27.
28.     printf( "The values of the modified array are:\n" );
29.
```

Функцийн загвар массив авч
байгааг зааж байна

Массив **a** -г **modifyArray** -д ЗӨВХӨН
нэрээр нь дамжуулж байна

Массивыг функцэд дамжуулах

```
30.  /* output modified array */
31.  for ( i = 0; i < SIZE; i++ ) {
32.      printf( "%3d", a[ i ] );
33.  } /* end for */
34.
35.  /* output value of a[ 3 ] */
36.  printf( "\n\n\nEffects of passing array element "
37.          "by value:\n\nThe value of a[3] is %d\n", a[ 3 ] );
38.
39.  modifyElement( a[ 3 ] ); /* pass array element a[ 3 ] by value */
40.
41.  /* output value of a[ 3 ] */
42.  printf( "The value of a[ 3 ] is %d\n", a[ 3 ] );
43.
44.  return 0; /* indicates successful termination */
45.
46. } /* end main */
47.
48. /* in function modifyArray, "b" points to the original array "a"
49.    in memory */
50. void modifyArray( int b[], int size )
51. {
52.     int j; /* counter */
53.
54.     /* multiply each array element by 2 */
55.     for ( j = 0; j < size; j++ ) {
56.         b[ j ] *= 2;
57.     } /* end for */
58.
59. } /* end function modifyArray */
```

Массивын элементийг
modifyElement -д утгаар
нь дамжуулж байна

Массивыг функцэд дамжуулах...

```
60.  
61. /* in function modifyElement, "e" is a local copy of array element  
62.    a[ 3 ] passed from main */  
63. void modifyElement( int e )  
64. {  
65.     /* multiply parameter by 2 */  
66.     printf( "Value in modifyElement is %d\n", e *= 2 );  
67. } /* end function modifyElement */
```

Effects of passing entire array by reference:

The values of the original array are:

0 1 2 3 4

The values of the modified array are:

0 2 4 6 8

Effects of passing array element by value:

The value of a[3] is 6

Value in modifyElement is 12

The value of a[3] is 6

Массивыг функцэд дамжуулах

```
1.  /* Ex_36 Demonstrating the const type qualifier with arrays */
2.  #include <stdio.h>
3.
4.  void tryToModifyArray( const int b[] ); /* function prototype */
5.
6.  /* function main begins program execution */
7.  int main( void )
8.  {
9.      int a[] = { 10, 20, 30 }; /* initialize a */
10.
11.     tryToModifyArray( a );
12.
13.     printf("%d %d %d\n", a[ 0 ], a[ 1 ], a[ 2 ] );
14.
15.     return 0; /* indicates successful termination */
16.
17. } /* end main */
18.
19. /* in function tryToModifyArray, array b is const, so it cannot be
20.    used to modify the original array a in main. */
21. void tryToModifyArray( const int b[] )
22. {
23.     b[ 0 ] /= 2; /* error */
24.     b[ 1 ] /= 2; /* error */
25.     b[ 2 ] /= 2; /* error */
26. } /* end function tryToModifyArray */
```

const тодорхойлогч массив
өөрчлөгдөхгүй гэдгийг
хөрвүүлэгчид хэлж өгч байна

Массивыг өөрчлөх аливаа
оролдлого алдаанд хүргэнэ

Массивыг функцэд дамжуулах...

Compiling...

example.c

example.c(23) : error C2166: l-value specifies const object

example.c(24) : error C2166: l-value specifies const object

example.c(25) : error C2166: l-value specifies const object

Лекцийн агуулга

- ▶ Массивыг тодорхойлох, идэвхижүүлэх, элементэд хандах
- ▶ Тэмдэгт тогтмолыг тодорхойлох
- ▶ Массивыг функцэд дамжуулах
- ▶ Жагсаалт, хүснэгтийн хадгалалт, эрэмбэлэлт, хайлтанд массивыг ашиглах
- ▶ Олон хэмжээст массивыг тодорхойлох, ашиглах
- ▶ Дүгнэлт

Массивын эрэмбэлэлт

► Өгөгдлийн эрэмбэлэлт

- Тооцооллын чухал хэрэглээ
- Байгууллага бүрт ямар нэг өгөгдлийг эрэмбэлэх хэрэгтэй болдог

► Бөмбөлгөн (bubble sort) эрэмбэлэлт

- Массив дотор дамжуулалт явагдана
- Зэргэлдээ элементүүдийг харьцуулна
 - Хэрвээ өсөх дараалалтай (эсхүл адилхан) бол өөрчлөхгүй
 - Хэрвээ буурах дараалалтай бол элементүүд солигдоно
- Давталт

► Жишээ

3	4	2	6	7
3	2	4	6	7
2	3	4	6	7

Массивыг функцэд дамжуулах

```
1.  /* Ex_37 This program sorts an array's values into ascending order */
2.  #include <stdio.h>
3.  #define SIZE 10
4.
5.  /* function main begins program execution */
6.  int main( void )
7.  {
8.      /* initialize a */
9.      int a[ SIZE ] = { 2, 6, 4, 8, 10, 12, 89, 68, 45, 37 };
10.     int pass; /* passes counter */
11.     int i;     /* comparisons counter */
12.     int hold; /* temporary location used to swap array elements */
13.
14.     printf( "Data items in original order\n" );
15.
16.     /* output original array */
17.     for ( i = 0; i < SIZE; i++ ) {
18.         printf( "%4d", a[ i ] );
19.     } /* end for */
20.
21.     /* bubble sort */
22.     /* loop to control number of passes */
23.     for ( pass = 1; pass < SIZE; pass++ ) {
24.
25.         /* loop to control number of comparisons per pass */
26.         for ( i = 0; i < SIZE - 1; i++ ) {
27.
```

Массивыг функцэд дамжуулах

```
29.      /* compare adjacent elements and swap them if first
30.      element is greater than second element */
31.      if ( a[ i ] > a[ i + 1 ] ) {
32.          hold = a[ i ];
33.          a[ i ] = a[ i + 1 ];
34.          a[ i + 1 ] = hold;
35.      } /* end if */
36.
37.  } /* end inner for */
38.
39. } /* end outer for */
40.
41. printf( "\nData items in ascending order\n" );
42.
43. /* output sorted array */
44. for ( i = 0; i < SIZE; i++ ) {
45.     printf( "%4d", a[ i ] );
46. } /* end for */
47.
48. printf( "\n" );
49.
50. return 0; /* indicates successful termination */
51.
52. } /* end main */
```

Массивын аливаа зэрэгцсэн хоёр
элементийн дараалал буруу бол солино

Data items in original order

2 6 4 8 10 12 89 68 45 37

Data items in ascending order

2 4 6 8 10 12 37 45 68 89

Массивын хайлт

- ▶ Массиваас *түлхүүр* утга хайх
- ▶ Шугаман хайлт
 - ▶ Энгийн
 - ▶ Массивын элемент бүрийг түлхүүр утгатай харьцуулна
 - ▶ Жижиг, эрэмбэлээгүй массивт тохиромжтой
- ▶ Хоёртын хайлт
 - ▶ Эрэмбэлэгдсэн массивт ашигладаг
 - ▶ Голын элементийг түлхүүртэй харьцуулна
 - ▶ Хэрвээ тэнцүү бол хайснаа оллоо гэж үзнэ
 - ▶ Хэрвээ *түлхүүр* < *голынх* бол массивын эхний хагасыг үзнэ
 - ▶ Хэрвээ *түлхүүр* > *голынх* бол массивын сүүлийн хагасыг үзнэ
 - ▶ Давталт
 - ▶ Маш хурдан(хэрвээ $2^n >$ элементийн тоо бол хамгийн ихдээ n алхам,. Жишээ: 30 элементтэй бол 5 алхам. Учир нь $2^5 > 30$)

Массивын шугаман хайлт

```
1.  /* Ex_38 Linear search of an array */
2.  #include <stdio.h>
3.  #define SIZE 100
4.
5.  /* function prototype */
6.  int linearSearch( const int array[], int key, int size );
7.
8.  /* function main begins program execution */
9.  int main( void )
10. {
11.     int a[ SIZE ]; /* create array a */
12.     int x; /* counter for initializing elements 0-99 of array a */
13.     int searchKey; /* value to locate in array a */
14.     int element; /* variable to hold location of searchKey or -1 */
15.
16.     /* create data */
17.     for ( x = 0; x < SIZE; x++ ) {
18.         a[ x ] = 2 * x;
19.     } /* end for */
20.
```

Массивын шугаман хайлт...

```
21.    printf( "Enter integer search key:\n" );
22.    scanf( "%d", &searchKey );
23.
24.    /* attempt to locate searchKey in array a */
25.    element = linearSearch( a, searchKey, SIZE );
26.
27.    /* display results */
28.    if ( element != -1 ) {
29.        printf( "Found value in element %d\n", element );
30.    } /* end if */
31.    else {
32.        printf( "Value not found\n" );
33.    } /* end else */
34.
35.    return 0; /* indicates successful termination */
36.
37. } /* end main */
38.
39. /* compare key to every element of array until the location is found
40.    or until the end of array is reached; return subscript of element
41.    if key or -1 if key is not found */
42. int linearSearch( const int array[], int key, int size )
43. {
44.     int n; /* counter */
45.
```

Массивын шугаман хайлт...

```
46.  /* loop through array */
47.  for ( n = 0; n < size; ++n ) {
48.
49.      if ( array[ n ] == key ) {
50.          return n; /* return location of key */
51.      } /* end if */
52.
53.  } /* end for */
54.
55.  return -1; /* key not found */
56.
57. } /* end function linearSearch */
```

Шугаман хайлтын алгоритм
массивын элемент бүрийг
шалгаж олтлоо хайдаг

Enter integer search key:

36

Found value in element 18

Enter integer search key:

37

Value not found

Массивын хоёртын хайлт

```
1.  /* Ex_39 Binary search of an array */
2.  #include <stdio.h>
3.  #define SIZE 15
4.
5.  /* function prototypes */
6.  int binarySearch( const int b[], int searchKey, int low, int high );
7.  void printHeader( void );
8.  void printRow( const int b[], int low, int mid, int high );
9.
10. /* function main begins program execution */
11. int main( void )
12. {
13.     int a[ SIZE ]; /* create array a */
14.     int i; /* counter for initializing elements 0-14 of array a */
15.     int key; /* value to locate in array a */
16.     int result; /* variable to hold location of key or -1 */
17.
18.     /* create data */
19.     for ( i = 0; i < SIZE; i++ ) {
20.         a[ i ] = 2 * i;
21.     } /* end for */
22.
23.     printf( "Enter a number between 0 and 28: " );
24.     scanf( "%d", &key );
25.
26.     printHeader();
27.
28.     /* search for key in array a */
29.     result = binarySearch( a, key, 0, SIZE - 1 );
```

Массивын хоёртын хайлт...

```
30.  /* display results */
31.  if ( result != -1 ) {
32.      printf( "\n%d found in array element %d\n", key, result );
33.  } /* end if */
34.  else {
35.      printf( "\n%d not found\n", key );
36.  } /* end else */
37.
38.  return 0; /* indicates successful termination */
39.
40. } /* end main */
41.
42. /* function to perform binary search of an array */
43. int binarySearch( const int b[], int searchKey, int low, int high )
44. {
45.     int middle; /* variable to hold middle element of array */
46.
47.     /* loop until low subscript is greater than high subscript */
48.     while ( low <= high ) {
49.
50.         /* determine middle element of subarray being searched */
51.         middle = ( low + high ) / 2;
52.
53.         /* display subarray used in this loop iteration */
54.         printRow( b, low, middle, high );
55.
```


Массивын хоёртын хайлт...

```
56.  /* if searchKey matched middle element, return middle */
57.  if ( searchKey == b[ middle ] ) {
58.      return middle;
59.  } /* end if */
60.
61.  /* if searchKey less than middle element, set new high */
62.  else if ( searchKey < b[ middle ] ) {
63.      high = middle - 1; /* search low end of array */
64.  } /* end else if */
65.
66.  /* if searchKey greater than middle element, set new low */
67.  else {
68.      low = middle + 1; /* search high end of array */
69.  } /* end else */
70.
71.  } /* end while */
72.
73.  return -1; /* searchKey not found */
74.
75. } /* end function binarySearch */
76.
77. /* Print a header for the output */
78. void printHeader( void )
79. {
80.     int i; /* counter */
81.
82.     printf( "\nSubscripts:\n" );
83.
```

Утга олдвол индексийг буцаана

Утга их бол хайлтыг зүүн хагаст хийнэ

Утга бага бол хайлтыг баруун хагаст хийнэ

Массивын хоёртын хайлт...

```
84.    /* output column head */
85.    for ( i = 0; i < SIZE; i++ ) {
86.        printf( "%3d ", i );
87.    } /* end for */
88.
89.    printf( "\n" ); /* start new line of output */
90.
91.    /* output line of - characters */
92.    for ( i = 1; i <= 4 * SIZE; i++ ) {
93.        printf( "-" );
94.    } /* end for */
95.
96.    printf( "\n" ); /* start new line of output */
97. } /* end function printHeader */
98.
99. /* Print one row of output showing the current
100. part of the array being processed. */
101. void printRow( const int b[], int low, int mid, int high )
102. {
103.     int i; /* counter for iterating through array b */
104.
```

Массивын хоёртын хайлт...

```
105.  /* loop through entire array */
106.  for ( i = 0; i < SIZE; i++ ) {
107.
108.      /* display spaces if outside current subarray range */
109.      if ( i < low || i > high ) {
110.          printf( "      " );
111.      } /* end if */
112.      else if ( i == mid ) { /* display middle element */
113.          printf( "%3d*", b[ i ] ); /* mark middle value */
114.      } /* end else if */
115.      else { /* display other elements in subarray */
116.          printf( "%3d ", b[ i ] );
117.      } /* end else */
118.
119.  } /* end for */
120.
121.  printf( "\n" ); /* start new line of output */
122.} /* end function printRow */
```

Enter a number between 0 and 28: 25

Subscripts:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

0	2	4	6	8	10	12	14*	16	18	20	22	24	26	28
								16	18	20	22*	24	26	28
												24	26*	28
												24*		

25 not found

Массивын хоёртын хайлт...

Enter a number between 0 and 28: 8

Subscripts:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	2	4	6	8	10	12	14*	16	18	20	22	24	26	28
0	2	4	6*	8	10	12								
				8	10*	12								
				8*										

8 found in array element 4

Enter a number between 0 and 28: 8

Subscripts:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	2	4	6	8	10	12	14*	16	18	20	22	24	26	28
0	2	4	6*	8	10	12								

6 found in array element 3

Лекцийн агуулга

- ▶ Массивыг тодорхойлох, идэвхижүүлэх, элементэд хандах
- ▶ Тэмдэгт тогтмолыг тодорхойлох
- ▶ Массивыг функцэд дамжуулах
- ▶ Жагсаалт, хүснэгтийн хадгалалт, эрэмбэлэлт, хайлтанд массивыг ашиглах
- ▶ Олон хэмжээст массивыг тодорхойлох, ашиглах
- ▶ Дүгнэлт

Олон хэмжээст массив

▶ Олон хэмжээст массив

- ▶ Мөр, баганаар илэрхийлэгдсэн хүснэгт ($m \times n$ массив)
- ▶ Матрицтай төстэй: мөрийн индексийн ард баганых байна

▶ Идэвхижүүлэлт

- ▶ Мөрөөр нь бүлэглэж, идэвхижүүлнэ

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

- ▶ Үлдсэн элемент тэг утгатай болно

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

▶ Элементэд хандах

- ▶ Мөр, баганыг заана

```
printf("%d", b[0][1]);
```

- ▶ **$a[x, y]$ – гэж бичихгүй, $a[x][y]$ – гэж болно**

Олон хэмжээст массив

	Багана 0	Багана 1	Багана 2	Багана 3
Мөр 0	<code>a[0][0]</code>	<code>a[0][1]</code>	<code>a[0][2]</code>	<code>a[0][3]</code>
Мөр 1	<code>a[1][0]</code>	<code>a[1][1]</code>	<code>a[1][2]</code>	<code>a[1][3]</code>
Мөр 2	<code>a[2][0]</code>	<code>a[2][1]</code>	<code>a[2][2]</code>	<code>a[2][3]</code>

Баганын индекс
Мөрийн индекс
Массивын нэр

Олон ХЭМЖЭЭСТ МАССИВ

```
1.  /* Ex_40 Initializing multidimensional arrays */
2.  #include <stdio.h>
3.
4.  void printArray( const int a[][ 3 ] ); /* function prototype */
5.
6.  /* function main begins program execution */
7.  int main( void )
8.  {
9.      /* initialize array1, array2, array3 */
10.     int array1[ 2 ][ 3 ] = { { 1, 2, 3 }, { 4, 5, 6 } };
11.     int array2[ 2 ][ 3 ] = { 1, 2, 3, 4, 5 };
12.     int array3[ 2 ][ 3 ] = { { 1, 2 }, { 4 } };
13.
14.     printf( "Values in array1 by row are:\n" );
15.     printArray( array1 );
16.
17.     printf( "Values in array2 by row are:\n" );
18.     printArray( array2 );
19.
20.     printf( "Values in array3 by row are:\n" );
21.     printArray( array3 );
22.
23.     return 0; /* indicates successful termination */
24.
25. } /* end main */
26.
```

Массив `array1` –н хоёр мөрийг бүхэлд нь идэвхижүүлж байна

Массив `array2, array3` хэсэгчилэн идэвхижүүлэгдэж байна

Олон ХЭМЖЭЭСТ МАССИВ...

```
27. /* function to output array with two rows and three columns */
28. void printArray( const int a[][ 3 ] )
29. {
30.     int i; /* row counter */
31.     int j; /* column counter */
32.
33.     /* loop through rows */
34.     for ( i = 0; i <= 1; i++ ) {
35.
36.         /* output column values */
37.         for ( j = 0; j <= 2; j++ ) {
38.             printf( "%d ", a[ i ][ j ] );
39.         } /* end inner for */
40.
41.         printf( "\n" ); /* start new line of output */
42.     } /* end outer for */
43.
44. } /* end function printArray */
```

Values in array1 by row are:

1 2 3

4 5 6

Values in array2 by row are:

1 2 3

4 5 0

Values in array3 by row are:

1 2 0

4 0 0

Олон ХЭМЖЭЭСТ МАССИВ

```
1.  /* Ex_41 Double-subscripted array example */
2.  #include <stdio.h>
3.  #define STUDENTS 3
4.  #define EXAMS 4
5.
6.  /* function prototypes */
7.  int minimum( const int grades[][ EXAMS ], int pupils, int tests );
8.  int maximum( const int grades[][ EXAMS ], int pupils, int tests );
9.  double average( const int setOfGrades[], int tests );
10. void printArray( const int grades[][ EXAMS ], int pupils, int tests );
11.
12. /* function main begins program execution */
13. int main( void )
14. {
15.     int student; /* student counter */
16.
17.     /* initialize student grades for three students (rows) */
18.     const int studentGrades[ STUDENTS ][ EXAMS ] =
19.         { { 77, 68, 86, 73 },
20.           { 96, 87, 89, 78 },
21.           { 70, 90, 86, 81 } };
22.
```

Массивын мөр бүр нь нэг оюутны
авсан шалгалтын оноонууд

Олон ХЭМЖЭЭСТ МАССИВ...

```
23.  /* output array studentGrades */
24.  printf( "The array is:\n" );
25.  printArray( studentGrades, STUDENTS, EXAMS );
26.
27.  /* determine smallest and largest grade values */
28.  printf( "\n\nLowest grade: %d\nHighest grade: %d\n",
29.          minimum( studentGrades, STUDENTS, EXAMS ),
30.          maximum( studentGrades, STUDENTS, EXAMS ) );
31.
32.  /* calculate average grade for each student */
33.  for ( student = 0; student < STUDENTS; student++ ) {
34.      printf( "The average grade for student %d is %.2f\n",
35.              student, average( studentGrades[ student ], EXAMS ) );
36.  } /* end for */
37.
38.  return 0; /* indicates successful termination */
39.
40. } /* end main */
41.
```

Массивын мөрийг **average** функцэд дамжуулж байна

Олон ХЭМЖЭЭСТ МАССИВ...

```
42. /* Find the minimum grade */
43. int minimum( const int grades[][ EXAMS ], int pupils, int tests )
44. {
45.     int i; /* student counter */
46.     int j; /* exam counter */
47.     int lowGrade = 100; /* initialize to highest possible grade */
48.
49.     /* loop through rows of grades */
50.     for ( i = 0; i < pupils; i++ ) {
51.
52.         /* loop through columns of grades */
53.         for ( j = 0; j < tests; j++ ) {
54.
55.             if ( grades[ i ][ j ] < lowGrade ) {
56.                 lowGrade = grades[ i ][ j ];
57.             } /* end if */
58.
59.         } /* end inner for */
60.
61.     } /* end outer for */
62.
63.     return lowGrade; /* return minimum grade */
64.
65. } /* end function minimum */
66.
```

Олон ХЭМЖЭЭСТ МАССИВ...

```
67. /* Find the maximum grade */
68. int maximum( const int grades[][ EXAMS ], int pupils, int tests )
69. {
70.     int i; /* student counter */
71.     int j; /* exam counter */
72.     int highGrade = 0; /* initialize to lowest possible grade */
73.
74.     /* loop through rows of grades */
75.     for ( i = 0; i < pupils; i++ ) {
76.
77.         /* loop through columns of grades */
78.         for ( j = 0; j < tests; j++ ) {
79.
80.             if ( grades[ i ][ j ] > highGrade ) {
81.                 highGrade = grades[ i ][ j ];
82.             } /* end if */
83.
84.         } /* end inner for */
85.
86.     } /* end outer for */
87.
88.     return highGrade; /* return maximum grade */
89.
90. } /* end function maximum */
91.
```

Олон ХЭМЖЭЭСТ МАССИВ...

```
92. /* Determine the average grade for a particular student */
93. double average( const int setOfGrades[], int tests )
94. {
95.     int i; /* exam counter */
96.     int total = 0; /* sum of test grades */
97.
98.     /* total all grades for one student */
99.     for ( i = 0; i < tests; i++ ) {
100.         total += setOfGrades[ i ];
101.     } /* end for */
102.
103.     return ( double ) total / tests; /* average */
104.
105. } /* end function average */
106.
107. /* Print the array */
108. void printArray( const int grades[][ EXAMS ], int pupils, int tests )
109. {
110.     int i; /* student counter */
111.     int j; /* exam counter */
112.
113.     /* output column heads */
114.     printf( "          [0]   [1]   [2]   [3]" );
115.
```

Олон ХЭМЖЭЭСТ МАССИВ...

```
116.    /* output grades in tabular format */
117.    for ( i = 0; i < pupils; i++ ) {
118.
119.        /* output label for row */
120.        printf( "\nstudentGrades[%d] ", i );
121.
122.        /* output grades for one student */
123.        for ( j = 0; j < tests; j++ ) {
124.            printf( "%-5d", grades[ i ][ j ] );
125.        } /* end inner for */
126.
127.    } /* end outer for */
128.
129.} /* end function printArray */
```

The array is:

	[0]	[1]	[2]	[3]
studentGrades[0]	77	68	86	73
studentGrades[1]	96	87	89	78
studentGrades[2]	70	90	86	81

Lowest grade: 68

Highest grade: 96

The average grade for student 0 is 76.00

The average grade for student 1 is 87.50

The average grade for student 2 is 81.75

Лекцийн агуулга

- ▶ Массивыг тодорхойлох, идэвхижүүлэх, элементэд хандах
- ▶ Тэмдэгт тогтмолыг тодорхойлох
- ▶ Массивыг функцэд дамжуулах
- ▶ Жагсаалт, хүснэгтийн хадгалалт, эрэмбэлэлт, хайлтанд массивыг ашиглах
- ▶ Олон хэмжээст массивыг тодорхойлох, ашиглах
- ▶ Дүгнэлт

Дүгнэлт

- ▶ Массвыг тодорхойлох/зарлахдаа

```
arrayType arrayName[numberOfElements];
```

- ▶ Тэмдэгт тогтмол

```
#define SIZE 12 /* no semicolon here!!! */
```

- ▶ Массивын хэмжээ

- ▶ Массивын индекс нь ≥ 0 ба ямагт массивын элементийн тооноос бага байна (size-1)

- ▶ Массивыг функцэд дамжуулах: **хаягаар-дуудах**

- ▶ Массивын эрэмбэлэлт: бөмбөлгөн эрэмбэлэлт

- ▶ Массивын хайлт: шугаман ба хоёртын хайлт

- ▶ Олон хэмжээст массив