

CS200 – Програмчлалын үндэс

Лекц 07

Мөр ба тэмдэгт.

Профессор А.Эрдэнэбаатар

Лекцийн агуулга

- ▶ Удиртгал
- ▶ Мөр, тэмдэгтийн үндэс
- ▶ Тэмдэгт боловсруулах сан
- ▶ Мөр хөрвүүлэгч функцүүд
- ▶ Стандарт оролт/гаралтын сангийн функцүүд
- ▶ Мөр боловсруулах сангийн мөртэй ажилладаг функцүүд
- ▶ Хэвшүүлсэн оролт/гаралт

Удиртгал

- ▶ Стандарт сангийн функцийн зарим танилцуулга
 - ▶ Мөр, тэмдэгтийг боловсруулахад хялбар
 - ▶ Програм тэмдэгт, мөр, олон мөр текст, санах ойн блокыг боловсруулж чадна
- ▶ Эдгээр техникийг ашиглаж
 - ▶ Текст процессор
 - ▶ Хуудасны зохиомжийн програм хангамж
 - ▶ Бичиг өрөлтийн програм зэргийг бүтээж чадна

Мөр, тэмдэгтийн үндэс

▶ Тэмдэгт

▶ Програмын үндсэн блок

- ▶ Аливаа програм бүр утгатай бүлэглэсэн тэмдэгтүүдийн цуваа байдаг

▶ Тэмдэгт тогтмол

- ▶ Дан хашилтанд байгаа тэмдэгтийн бүхэл утга
- ▶ Тухайлбал `'z'` нь `z` –ийн бүхэл утгын дүрслэл

▶ Мөр

▶ Нэг бүхэл гэж харьцах цуварсан тэмдэгтүүд

- ▶ Үсэг, цифр, тусгай тэмдэгтүүд байж болно (`*`, `/`, `$`)

▶ Мөр (тогтмол) нь давхар хашилтанд бичигдэнэ: `"Hello"`

▶ Мөр бол тэмдэгтүүдийн массив

- ▶ Мөр нь эхний тэмдэгтийн заагч, утга нь эхний тэмдэгтийн хаяг

Мөр, тэмдэгтийн үндэс

▶ Мөрийг тодорхойлох/зарлах

- ▶ Тэмдэгтийн массив буюу `char *` төрлийн хувьсагч байдлаар тодорхойлогдоно

```
char color[] = "blue";
char *colorPtr = "blue";
```
- ▶ Мөр нь `'\0'` –аар төгссөн тэмдэгт массиваар дүрслэгддэгийг санаарай. (`color` нь 5 элементтэй)

▶ Мөрийг оруулах

- ▶ `scanf` –г ашиглах: `scanf ("%s", word) ;`
 - ▶ Оролтыг `word[]` –д хуулна
 - ▶ `&` -г ашиглах шаардлагагүй (учир нь мөр бол заагч)
- ▶ Массивдаа `'\0'` –д зай үлдээхээ мартаж болохгүй

Лекцийн агуулга

- ▶ Удиртгал
- ▶ Мөр, тэмдэгтийн үндэс
- ▶ Тэмдэгт боловсруулах сан
- ▶ Мөр хөрвүүлэгч функцүүд
- ▶ Стандарт оролт/гаралтын сангийн функцүүд
- ▶ Мөр боловсруулах сангийн мөртэй ажилладаг функцүүд
- ▶ Хэвшүүлсэн оролт/гаралт

Тэмдэгт боловсруулах сан

- ▶ Тэмдэгт боловсруулах сан
 - ▶ Тэмдэгт өгөгдөлтэй ажиллах, шалгахад хэрэгтэй функцүүдийг багтаасан
 - ▶ Функц бүр тэмдэгт (бүхэл), эсхүл `EOF` –г аргумент болгож авдаг
- ▶ Дараачийн слайд `<ctype.h>` -д орсон тэмдэгт боловсруулах сангийн функцүүдийг хүснэгтээр харуулна

Тэмдэгт боловсруулах сангийн функцүүд

Загвар	Тайлбар
<code>int isdigit(int c);</code>	Цифр бол true; бусад тохиолдолд false буцна
<code>int isalpha(int c);</code>	Үсэг бол true; бусад тохиолдолд false буцна
<code>int isalnum(int c);</code>	Цифр, үсэг бол true; бусад тохиолдолд false буцна
<code>int isxdigit(int c);</code>	16 –тын цифр бол true; бусад тохиолдолд false буцна
<code>int islower(int c);</code>	Жижиг үсэг бол true; бусад тохиолдолд false буцна
<code>int isupper(int c);</code>	Том үсэг бол true; бусад тохиолдолд false буцна
<code>int tolower(int c);</code>	Жижиг үсэг болгож буцаана
<code>int toupper(int c);</code>	Том үсэг болгож буцаана
<code>int isspace(int c);</code>	Цагаан-зай тэмдэгт бол true; бусад тохиолдолд false буцна. Цагаан-зай: ' ', '\n', '\f', '\r', '\t', '\v'
<code>int iscntrl(int c);</code>	Удирдлагын тэмдэгт бол true; бусад тохиолдолд false буцна
<code>int ispunct(int c);</code>	Зай, үсэг, цифрээс бусад хэвлэгдэх тэмдэгт бол true; бусад тохиолдолд false буцна
<code>int isprint(int c);</code>	Хэвлэгдэх тэмдэгт бол true; бусад тохиолдолд false буцна
<code>int isgraph(int c);</code>	Зайнаас бусад хэвлэгдэх тэмдэгт бол true; бусад тохиолдолд false буцна

Лекцийн агуулга

- ▶ Удиртгал
- ▶ Мөр, тэмдэгтийн үндэс
- ▶ Тэмдэгт боловсруулах сан
- ▶ Мөр хөрвүүлэгч функцүүд
- ▶ Стандарт оролт/гаралтын сангийн функцүүд
- ▶ Мөр боловсруулах сангийн мөртэй ажилладаг функцүүд
- ▶ Хэвшүүлсэн оролт/гаралт

Мөр хөрвүүлэгч функцүүд

- ▶ Хөрвүүлэгч функцүүд
 - ▶ `<stdlib.h>` -д тодорхойлогдсон(нийтлэг хэрэгслийн сан)
- ▶ Цифрээс бүтсэн мөрийг бүхэл болон хөвөгч таслалтай утга болгож хөрвүүлнэ

Загвар	Тайлбар
<code>double atof(const char *nPtr);</code>	Мөрийг double –д хөрвүүлнэ
<code>int atoi(const char *nPtr);</code>	Мөрийг int –д хөрвүүлнэ
<code>long atol(const char *nPtr, char **endPtr);</code>	Мөрийг long int –д хөрвүүлнэ
<code>double strtod(const char *nPtr, char **endPtr);</code>	Мөрийг double –д хөрвүүлнэ
<code>long strtol(const char *nPtr, char **endPtr, int base);</code>	Мөрийг long –д хөрвүүлнэ
<code>unsigned long strtoul(const char *nPtr, char **endPtr, int base);</code>	Мөрийг unsigned long –д хөрвүүлнэ

Мөр хөрвүүлэгч функцүүд

```
1.  /* Ex_57 Using atof */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.
5.  int main( void )
6.  {
7.      double d; /* variable to hold converted string */
8.
9.      d = atof( "99.0" );
10.
11.     printf( "%s%.3f\n%s%.3f\n",
12.             "The string \"99.0\" converted to double is ", d,
13.             "The converted value divided by 2 is ",
14.             d / 2.0 );
15.
16.     return 0; /* indicates successful termination */
17.
18. } /* end main */
```

The string "99.0" converted to double is 99.000
The converted value divided by 2 is 49.500

atof нь мөрийг double -д хөрвүүлнэ

Санамж: Хэрвээ мөрний эхний тэмдэгт нь үсэг бол `atof` функц яаж ажиллах нь тодорхойгүй

Мөр хөрвүүлэгч функцүүд

```
1.  /* Ex_58 Using atoi */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.
5.  int main( void )
6.  {
7.      int i; /* variable to hold converted string */
8.
9.      i = atoi( "2593" );
10.
11.     printf( "%s%d\n%s%d\n",
12.             "The string \"2593\" converted to int is ", i,
13.             "The converted value minus 593 is ", i - 593 );
14.
15.     return 0; /* indicates successful termination */
16.
17. } /* end main */
```

The string "2593" converted to int is 2593
The converted value minus 593 is 2000

Мөр хөрвүүлэгч функцүүд

```
1.  /* Ex_59 Using atol */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.
5.  int main( void )
6.  {
7.      long l; /* variable to hold converted string */
8.
9.      l = atol( "1000000" );
10.
11.     printf( "%s%ld\n%s%ld\n",
12.             "The string \"1000000\" converted to long int is ", l,
13.             "The converted value divided by 2 is ", l / 2 );
14.
15.     return 0; /* indicates successful termination */
16.
17. } /* end main */
```

The string "1000000" converted to long int is 1000000
The converted value divided by 2 is 500000

← atol нь мөрийг long -д хөрвүүлнэ

Мөр хөрвүүлэгч функцүүд

```
1.  /* Ex_60 Using strtod */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.
5.  int main( void )
6.  {
7.      /* initialize string pointer */
8.      const char *string = "51.2% are admitted"; /* initialize string */
9.
10.     double d;          /* variable to hold converted sequence */
11.     char *stringPtr; /* create char pointer */
12.
13.     d = strtod( string, &stringPtr );
14.
15.     printf( "The string \"%s\" is converted to the\n", string );
16.     printf( "double value %.2f and the string \"%s\"\n", d, stringPtr );
17.
18.     return 0; /* indicates successful termination */
19.
20. } /* end main */
```

The string "51.2 are admitted" is converted to the double value 51.20 and the string "% are admitted"

strtod нь мөрийн хэсгийг double -
д хөрвүүлнэ

Мөр хөрвүүлэгч функцүүд

```
1.  /* Ex_61 Using strtol */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.
5.  int main( void )
6.  {
7.      const char *string = "-1234567abc"; /* initialize string pointer */
8.
9.      char *remainderPtr; /* create char pointer */
10.     long x;              /* variable to hold converted sequence */
11.
12.     x = strtol( string, &remainderPtr, 0 );
13.
14.     printf( "%s\\\"%s\\\"\\n%s%ld\\n%s\\\"%s\\\"\\n%s%ld\\n",
15.             "The original string is ", string,
16.             "The converted value is ", x,
17.             "The remainder of the original string is ",
18.             remainderPtr,
19.             "The converted value plus 567 is ", x + 567 );
20.
21.     return 0; /* indicates successful termination */
22.
23. } /* end main */
```

strtod нь мөрийн хэсгийг long-д хөрвүүлнэ

Суурь.

Санамж:

1. Хоёр дахь параметрын оронд NULL –г ашиглавал мөрийн үлдэх хэсгийг алгасна
2. Гурав дахь параметр 8, 10, 16 –тын суурийг заана

```
The original string is "-1234567abc"
The converted value is -1234567
The remainder of the original string is "abc"
The converted value plus 567 is -1234000
```

Мөр хөрвүүлэгч функцүүд

```
1.  /* Ex_62 Using strtoul */
2.  #include <stdio.h>
3.  #include <stdlib.h>
4.
5.  int main( void )
6.  {
7.      const char *string = "1234567abc"; /* initialize string pointer */
8.      unsigned long x; /* variable to hold converted sequence */
9.      char *remainderPtr; /* create char pointer */
10.
11.     x = strtoul( string, &remainderPtr, 0 );
12.
13.     printf( "%s\\\"%s\\\"\\n%s%lu\\n%s\\\"%s\\\"\\n%s%lu\\n",
14.             "The original string is ", string,
15.             "The converted value is ", x,
16.             "The remainder of the original string is ",
17.             remainderPtr,
18.             "The converted value minus 567 is ", x - 567 );
19.
20.     return 0; /* indicates successful termination */
21.
22. } /* end main */
```

strtod нь мөрийн хэсгийг
unsigned long -д хөрвүүлнэ

```
The original string is "1234567abc"
The converted value is 1234567
The remainder of the original string is "abc"
The converted value plus 567 is 1234000
```


Лекцийн агуулга

- ▶ Удиртгал
- ▶ Мөр, тэмдэгтийн үндэс
- ▶ Тэмдэгт боловсруулах сан
- ▶ Мөр хөрвүүлэгч функцүүд
- ▶ Стандарт оролт/гаралтын сангийн функцүүд
- ▶ Мөр боловсруулах сангийн мөртэй ажилладаг функцүүд
- ▶ Хэвшүүлсэн оролт/гаралт

Стандарт оролт/гаралтын сангийн функцүүд

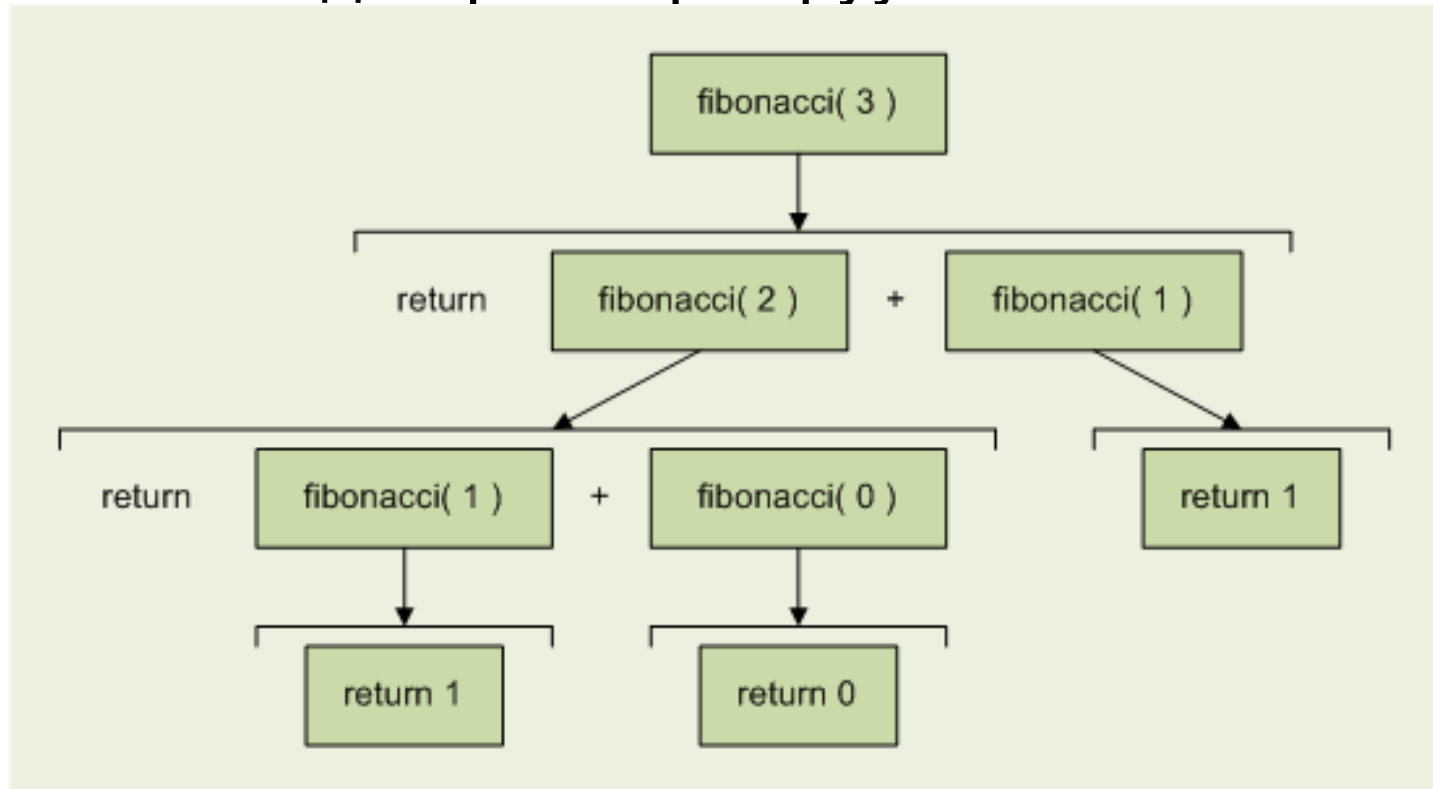
- ▶ `<stdio.h>` -д тодорхойлогдсон
- ▶ Мөр болон тэмдэгт өгөгдөлтэй ажилладаг

Загвар	Тайлбар
<code>int getchar(void);</code>	Стандарт оролтоос дараачийн тэмдэгтийг авч бүхэл болгон буцаана
<code>char *gets(char *s);</code>	Стандарт оролтоос тэмдэгтүүдийг шинэ мөр, файлын төгсгөл гартал авч массивт хийнэ. Төгсгөлд нь <code>NULL</code> тэмдэгтийг нэмнэ. Оруулсан мөрийг буцаана. Массив хангалтгүй бол алдаа гарна.
<code>int putchar(int c);</code>	Тэмдэгтийг хэвлэж бүхэл болгон буцаана
<code>int puts(const char *s);</code>	Мөрийг шинэ мөрийн тэмдэгттэй хэвлэнэ. Амжилттай бол 0 биш бүхлийг буцаана. Алдаа гарвал <code>EOF</code> –г буцаана
<code>int sprintf(char *s, const char *format, ...);</code>	<code>printf</code> –н эквивалент. Ялгаа нь гаралтаа массивт хадгална. Хадгалсан тэмдэгтийн тоог буцаана. Алдаа гарвал <code>EOF</code> –г буцаана
<code>int sscanf(char *s, const char *format, ...);</code>	<code>scanf</code> –н эквивалент. Ялгаа нь массиваас оруулна. Оруулсан зүйлийн тоог буцаана. Алдаа гарвал <code>EOF</code> –г буцаана

Стандарт оролт/гаралтын сангийн функцүүд

-Рекурсив функц

- ▶ Функц их бие дотроо өөрийгөө дуудна
- ▶ Суурь тохиолдол гартал дахин дахин хийгдэнэ
- ▶ Ажиллагааг диаграмаар харуулбал:



Стандарт оролт/гаралтын сангийн функцүүд

-Рекурсив функц...

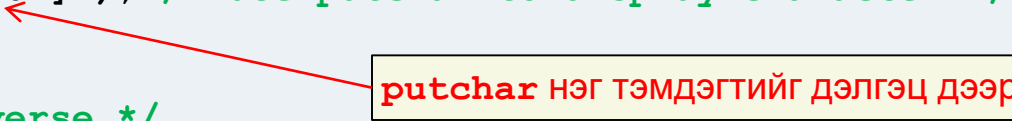
```
1.  /* Ex_63 Using gets and putchar */
2.  #include <stdio.h>
3.
4.  void reverse( const char * const sPtr ); /* prototype */
5.
6.  int main( void )
7.  {
8.      char sentence[ 80 ]; /* create char array */
9.
10.     printf( "Enter a line of text:\n" );
11.
12.     /* use gets to read line of text */
13.     gets( sentence );
14.
15.     printf( "\nThe line printed backward is:\n" );
16.     reverse( sentence );
17.
18.     return 0; /* indicates successful termination */
19.
20. } /* end main */
21.
```

gets хэрэглэгчээс текст мөрийг уншина

Стандарт оролт/гаралтын сангийн функцүүд

-Рекурсив функц...

```
21. /* recursively outputs characters in string in reverse order */
22. void reverse( const char * const sPtr )
23. {
24.     /* if end of the string */
25.     if ( sPtr[ 0 ] == '\0' ) { /* base case */
26.         return;
27.     } /* end if */
28.     else { /* if not end of the string */
29.         reverse( &sPtr[ 1 ] ); /* recursion step */
30.
31.         putchar( sPtr[ 0 ] ); /* use putchar to display character */
32.     } /* end else */
33.
34. } /* end function reverse */
```



Enter a line of text:

Characters and Strings

The line printed backward is:

sgnirtS dna sretcarahC

Enter a line of text:

able was I ere I saw elba

The line printed backward is:

able was I ere I saw elba

Стандарт оролт/гаралтын сангийн функцүүд - Рекурсив функц...Задлан шинжилгээ

- ▶ Алхам 1: `reverse("abc");`
 - ▶ Алхам 2: `reverse(&sPtr[1]) → reverse("bc")`(Эхлээд хий)
 - ▶ `putchar(sPtr[0]) → putchar('a')`(Дараа нь хий)
- ▶ Алхам 2: `reverse("bc");`
 - ▶ Алхам 3: `reverse(&sPtr[1]) → reverse("c")`(Эхлээд хий)
 - ▶ `putchar(sPtr[0]) → putchar('b')`(Дараа нь хий)
- ▶ Алхам 3: `reverse("c");`
 - ▶ Алхам 2: `reverse(&sPtr[1]) → reverse("\0")`(Эхлээд хий)
 - ▶ `putchar(sPtr[0]) → putchar('c')`(Дараа нь хий)
- ▶ Алхам 4: `reverse("\0");`
 - ▶ Суурь тохиолдол → `return`
- ▶ Алхам 4 → Алхам 3 → Алхам 2 → Алхам 1 "cba"

Стандарт оролт/гаралтын сангийн функцүүд

```
1.  /* Ex_64 Using getchar and puts */
2.  #include <stdio.h>
3.
4.  int main( void )
5.  {
6.      char c;                /* variable to hold character input by user */
7.      char sentence[ 80 ]; /* create char array */
8.      int i = 0;             /* initialize counter i */
9.
10.     /* prompt user to enter line of text */
11.     puts( "Enter a line of text:" );
12.
13.     /* use getchar to read each character */
14.     while ( ( c = getchar() ) != '\n' ) {
15.         sentence[ i++ ] = c;
16.     } /* end while */
17.
18.     sentence[ i ] = '\0'; /* terminate string */
19.
```

puts текст мөрийг дэлгэцэнд хэвлэнэ

Стандарт оролт/гаралтын сангийн функцүүд

```
20.  /* use puts to display sentence */
21.    puts( "\nThe line entered was:" );
22.    puts( sentence );
23.
24.    return 0; /* indicates successful termination */
25.
26. } /* end main */
```

Enter a line of text:

This is a test.

The line entered was:

This is a test.

Стандарт оролт/гаралтын сангийн функцүүд

```
1.  /* Ex_65 Using sprintf */
2.  #include <stdio.h>
3.
4.  int main( void )
5.  {
6.      char s[ 80 ]; /* create char array */
7.      int x;         /* x value to be input */
8.      double y;      /* y value to be input */
9.
10.     printf( "Enter an integer and a double:\n" );
11.     scanf( "%d%lf", &x, &y );
12.
13.     sprintf( s, "integer:%6d\ndouble:%8.2f", x, y );
14.
15.     printf( "%s\n%s\n",
16.             "The formatted output stored in array s is:", s );
17.
18.     return 0; /* indicates successful termination */
19.
20. } /* end main */
```

Enter an integer and a double:
298 87.375

The formatted output stored in a s is:
integer: 298
double: 87.38

sprintf нь текст мөрийг
printf дэлгэцэнд хэвлэдэг шиг
массивт хадгална

Стандарт оролт/гаралтын сангийн функцүүд

```
1.  /* Ex_66 Using sscanf */
2.  #include <stdio.h>
3.
4.  int main( void )
5.  {
6.      char s[] = "31298 87.375"; /* initialize array s */
7.      int x;    /* x value to be input */
8.      double y; /* y value to be input */
9.
10.     sscanf( s, "%d%lf", &x, &y );
11.
12.     printf( "%s\n%s%6d\n%s%8.3f\n",
13.             "The values stored in character array s are:",
14.             "integer:", x, "double:", y );
15.
16.     return 0; /* indicates successful termination */
17.
18. } /* end main */
```

sscanf текст мөрийг **scanf**
хэрэглэгчээс авдаг шиг
массиваас уншина

The value stored in character array s are:
integer: 31298
double: 87.375

Лекцийн агуулга

- ▶ Удиртгал
- ▶ Мөр, тэмдэгтийн үндэс
- ▶ Тэмдэгт боловсруулах сан
- ▶ Мөр хөрвүүлэгч функцүүд
- ▶ Стандарт оролт/гаралтын сангийн функцүүд
- ▶ Мөр боловсруулах сангийн мөртэй ажилладаг функцүүд
- ▶ Хэвшүүлсэн оролт/гаралт

Мөр боловсруулах сангийн мөртэй ажилладаг функцүүд

- ▶ Мөр боловсруулах сангийн функцүүдийг ашиглаж
 - ▶ Мөр өгөгдөлтэй ажиллах
 - ▶ Мөр хайх
 - ▶ Мөрийг задлах
 - ▶ Мөрийн уртыг олох зэргийг хийж болно

Загвар	Тайлбар
<code>char *strcpy(char *s1, const char *s2);</code>	s2 мөрийг s1 массивт хуулна. s1 –н утга буцна
<code>char *strncpy(char *s1, const char *s2, size_t n);</code>	s2 мөрөөс n тэмдэгтийг s1 массивт хуулна. s1 –н утга буцна
<code>char *strcat(char *s1, const char *s2);</code>	s2 мөрийг s1 массивт нэмнэ. s2 –н эхний тэмтэгт s1 –н 0 тэмдэгтийг дарна. s1 –н утга буцна
<code>char *strncat(char *s1, const char *s2, size_t n);</code>	s2 мөрөөс n тэмдэгтийг s1 массивт нэмнэ. s2 –н эхний тэмтэгт s1 –н 0 тэмдэгтийг дарна. s1 –н утга буцна

Мөртэй ажилладаг функцүүд

```
1.  /* Ex_67 Using strcpy and strncpy */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      char x[] = "Happy Birthday to You"; /* initialize char array x */
8.      char y[ 25 ]; /* create char array y */
9.      char z[ 15 ]; /* create char array z */
10.
11.     /* copy contents of x into y */
12.     printf( "%s%s\n%s%s\n",
13.         "The string in array x is: ", x,
14.         "The string in array y is: ", strcpy( y, x ) );
15.
16.     /* copy first 14 characters of x into z. Does not copy null
17.        character */
18.     strncpy( z, x, 14 );
19.
20.     z[ 14 ] = '\\0'; /* terminate string in z */
21.     printf( "The string in array z is: %s\n", z );
22.
23.     return 0; /* indicates successful termination */
24.
25. } /* end main */
```

strcpy мөр x -г тэмдэгт массив y -т хуулна

strncpy мөр x -н 14 тэмдэгтийг тэмдэгт массив z -т хуулна

strncpy нь 0 тэмдэгтийг автоматаар нэмэхгүй байгааг анхаар

```
The string in array x is: Happy Birthday to You
The string in array y is: Happy Birthday to You
The string in array z is: Happy Birthday
```

Мөртэй ажилладаг функцүүд

```
1.  /* Ex_68 Using strcat and strncat */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      char s1[ 20 ] = "Happy "; /* initialize char array s1 */
8.      char s2[] = "New Year "; /* initialize char array s2 */
9.      char s3[ 40 ] = "";      /* initialize char array s3 to empty */
10.
11.     printf( "s1 = %s\ns2 = %s\n", s1, s2 );
12.
13.     /* concatenate s2 to s1 */
14.     printf( "strcat( s1, s2 ) = %s\n", strcat( s1, s2 ) );
15.
16.     /* concatenate first 6 characters of s1 to s3. Place '\0'
17.        after last character */
18.     printf( "strncat( s3, s1, 6 ) = %s\n", strncat( s3, s1, 6 ) );
19.
```

strcat мөр **s2** -н тэмдэгтүүдийг
мөр **s1** -н ард нэмнэ

strncat нь **s1** -н 6
тэмдэгтийг мөр **s3** -н ард
нэмнэ. 0 тэмтэгт автоматаар
нэмэгдэнэ гэдгийг анхаар

Мөртэй ажилладаг функцүүд...

```
20. /* concatenate s1 to s3 */
21.     printf( "strcat( s3, s1 ) = %s\n", strcat( s3, s1 ) );
22.
23.     return 0; /* indicates successful termination */
24.
25. } /* end main */
s1 = Happy
s2 = New year
strcat( s1, s2 ) = Happy New Year
strncat( s3, s1, 6 ) = Happy
strcat( s3, s1 ) = Happy Happy New Year
```

Мөр боловсруулах сангийн харьцуулах функцүүд

► Мөрүүдийг харьцуулах

- Мөрийн тэмдэгтүүдийн ASCII тоон кодыг харьцуулдаг

```
int strcmp( const char *s1, const char *s2 );
```

- `s1` мөрийг `s2` мөртэй харьцуулна
- `s1 < s2` бол сөрөг, `s1 == s2` эсхүл `s1 > s2` бол эерэг тоог буцаана

```
int strncmp( const char *s1, const char *s2, size_t n );
```

- `s1` мөрийн `n` хүртэл тэмдэгтийг `s2` мөртэй харьцуулна
- `s1 < s2` бол сөрөг, `s1 == s2` эсхүл `s1 > s2` бол эерэг тоог буцаана

Мөртэй ажилладаг функцүүд

```
1.  /* Ex_69 Using strcmp and strncmp */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      const char *s1 = "Happy New Year"; /* initialize char pointer */
8.      const char *s2 = "Happy New Year"; /* initialize char pointer */
9.      const char *s3 = "Happy Holidays"; /* initialize char pointer */
10.
11.     printf("%s%s\n%s%s\n%s%s\n\n%s%2d\n%s%2d\n%s%2d\n\n",
12.           "s1 = ", s1, "s2 = ", s2, "s3 = ", s3,
13.           "strcmp(s1, s2) = ", strcmp( s1, s2 ),
14.           "strcmp(s1, s3) = ", strcmp( s1, s3 ),
15.           "strcmp(s3, s1) = ", strcmp( s3, s1 ) );
16.
```

strcmp мөр s1 -г мөр
s2 -той харьцуулна

Мөртэй ажилладаг функцүүд

```
17.     printf("%s%2d\n%s%2d\n%s%2d\n",
18.           "strncmp(s1, s3, 6) = ", strncmp( s1, s3, 6 ),
19.           "strncmp(s1, s3, 7) = ", strncmp( s1, s3, 7 ),
20.           "strncmp(s3, s1, 7) = ", strncmp( s3, s1, 7 ) );
21.
22.     return 0; /* indicates successful termination */
23.
24. } /* end main */
s1 = Happy New Year
s2 = Happy New Year
s3 = Happy Holidays

strcmp(s1, s2) = 0
strcmp(s1, s3) = 1
strcmp(s3, s1) = -1

strncmp(s1, s3, 6) = 0
strncmp(s1, s3, 7) = 1
strncmp(s3, s1, 7) = -1
```

strncmp мөр s1 –н эхний 6 тэмдэгтийг мөр s3 –н эхний 6 тэмдэгттэй харьцуулна

Мөр боловсруулах сангийн хайлтын функцүүд

Загвар	Тайлбар
<code>char *strchr(const char *s1, int c);</code>	s1 мөрөн дэх с тэмдэгтийн анхны илрэцийн олно. Олсон бол s1 дэх заагчийг буцаана. Эсрэг тохиолдолд NULL заагч буцна
<code>size_t strcspn(const char *s1, const char *s2);</code>	s2 мөрийн тэмдэгтээс мөр s1 –т оруугүй s1 –н эхний сегментийн уртыг буцаана. Ямарч тэмдэгт ороогүй бол s1 –н урт буцна
<code>size_t strspn(const char *s1, const char *s2);</code>	s2 мөрийн тэмдэгтээс мөр s1 –т орсон s1 –н эхний сегментийн уртыг буцаана. s1 –н бүх тэмдэгт s2 –т байгаа бол s1 –н урт буцна. s1 –н эхний тэмдэгт s2 –т байхгүй бол 0 буцна.
<code>char *strpbrk(const char *s1, const char *s2);</code>	s1 мөрөн дэх s2 мөрийн дурын тэмдэгтийн анхны илрэцийн олно. Олсон бол s1 дэх заагчийг буцаана. Эсрэг тохиолдолд NULL заагч буцна
<code>char *strrchr(const char *s1, int c);</code>	s1 мөрөн дэх с тэмдэгтийн сүүлчийн илрэцийн олно. Олсон бол s1 дэх заагчийг буцаана. Эсрэг тохиолдолд NULL заагч буцна
<code>char *strstr(const char *s1, const char *s2);</code>	s1 мөрөн дэх s2 мөрийн анхны илрэцийн олно. Олсон бол s1 дэх заагчийг буцаана. Эсрэг тохиолдолд NULL заагч буцна
<code>char *strtok(char *s1, const char *s2);</code>	s2 мөрийн тэмдэгтээр s1 мөрийг логик нэгж болгон задлана. Эхний удаад s1, цаашдаа NULL –г эхний аргумент болгоно. Дуудах тоолонд логик нэгжийн заагчийг буцаана. Дууссан бол NULL заагч буцна

Мөр боловсруулах сангийн хайлтын функцүүд

```
1.  /* Ex_70 Using strchr */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      const char *string = "This is a test"; /* initialize char pointer */
8.      char character1 = 'a'; /* initialize character1 */
9.      char character2 = 'z'; /* initialize character2 */
10.
11.     /* if character1 was found in string */
12.     if ( strchr( string, character1 ) != NULL ) {
13.         printf( "'%c' was found in \"%s\".\n",
14.             character1, string );
15.     } /* end if */
16.     else { /* if character1 was not found */
17.         printf( "'%c' was not found in \"%s\".\n",
18.             character1, string );
19.     } /* end else */
```

strchr мөр string-д
тэмдэгт character1 –
н эхний илрэцийг хайна

Мөр боловсруулах сангийн хайлтын функцүүд...

```
20.  
21.  /* if character2 was found in string */  
22.  if ( strchr( string, character2 ) != NULL ) {  
23.      printf( "'%c' was found in \"%s\".\n",  
24.          character2, string );  
25.  } /* end if */  
26.  else { /* if character2 was not found */  
27.      printf( "'%c' was not found in \"%s\".\n",  
28.          character2, string );  
29.  } /* end else */  
30.  
31.  return 0; /* indicates successful termination */  
32.  
33. } /* end main */  
'a' was found in "This is a test".  
'z' was not found in "This is a test".
```

Мөр боловсруулах сангийн хайлтын функцүүд

```
1.  /* Ex_71 Using strstr */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      const char *string1 = "abcdefabcdef"; /* string to search */
8.      const char *string2 = "def"; /* string to search for */
9.
10.     printf( "%s%s\n%s%s\n\n%s\n%s%s\n",
11.             "string1 = ", string1, "string2 = ", string2,
12.             "The remainder of string1 beginning with the",
13.             "first occurrence of string2 is: ",
14.             strstr( string1, string2 ) ); ←
15.
16.     return 0; /* indicates successful termination */
17.
18. } /* end main */
string1 = abcdefabcdef
string2 = def
```

strstr нь мөр string2 -н мөр string1 дэх эхний илрэцийн дараах үлдэгдлийг буцаана

The remainder of string1 beginning with the first occurrence of string2 is: defabcdef

Мөр боловсруулах сангийн хайлтын функцүүд

```
1.  /* Ex_72 Using strtok */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      /* initialize array string */
8.      char string[] = "This is a sentence with 7 tokens";
9.      char *tokenPtr; /* create char pointer */
10.
11.     printf( "%s\n%s\n\n%s\n",
12.             "The string to be tokenized is:", string,
13.             "The tokens are:" );
14.
15.     tokenPtr = strtok( string, " " ); /* begin tokenizing sentence */
16.
17.     /* continue tokenizing sentence until tokenPtr becomes NULL */
18.     while ( tokenPtr != NULL ) {
19.         printf( "%s\n", tokenPtr );
20.         tokenPtr = strtok( NULL, " " ); /* get next token */
21.     } /* end while */
```

strtok нь мөр **string** -г зай тэмдэгтээр нэгжүүд болгон задлана

strtok -г **NULL** аргументтэй дахин дуудсанаар задаргааны дараачийн нэгжийг гаргана

Мөр боловсруулах сангийн хайлтын функцүүд...

```
22.  
23.     return 0; /* indicates successful termination */  
24.  
25. } /* end main */
```

The string to be tokenized is:

This is a sentence with 7 tokens

The tokens are:

This

is

a

sentence

with

7

tokens

Мөр боловсруулах сангийн бусад функцүүд

Загвар	Тайлбар
<code>size_t strlen(const char *s);</code>	s мөрийн 0 тэмдэгтээс өмнөх уртыг буцаана.

Мөр боловсруулах сангийн бусад функцүүд

```
1.  /* Ex_73 Using strlen */
2.  #include <stdio.h>
3.  #include <string.h>
4.
5.  int main( void )
6.  {
7.      /* initialize 3 char pointers */
8.      const char *string1 = "abcdefghijklmnopqrstuvwxyz";
9.      const char *string2 = "four";
10.     const char *string3 = "Boston";
11.
12.     printf("%s\\\"%s\\\"%s%lu\\n%s\\\"%s\\\"%s%lu\\n%s\\\"%s\\\"%s%lu\\n",
13.           "The length of ", string1, " is ",
14.           ( unsigned long ) strlen( string1 ),
15.           "The length of ", string2, " is ",
16.           ( unsigned long ) strlen( string2 ),
17.           "The length of ", string3, " is ",
18.           ( unsigned long ) strlen( string3 ) );
19.
20.     return 0; /* indicates successful termination */
21.
22. } /* end main */
```

strlen нь мөр string1 -н
уртыг буцаана

The length of "abcdefghijklmnopqrstuvwxyz" is 26

The length of "four" is 4

The length of "Boston" is 6