# СТАНДАРТНАЯ БИБЛИОТЕКА СИ

Тема 1. Раздел 3. **stdlib.h**

# stdlib.h

- Преобразование типов
- Генерация псевдослучайных последовательностей
- Выделение и освобождение памяти
- Контроль процесса выполнения программы
- Сортировка и поиск
- Математика
- Многобайтовые операции/ широкие символы

# Преобразование типов

**double atof( const char *str );**

- Convert a string to double

**Return Value**

Each function returns the double value produced by interpreting the input characters as a number. The return value is 0.0 if the input cannot be converted to a value of that type.

```
int main ()
{
   double n,m;
   double pi=3.1415926535;
   char szInput [256];
   printf ( "Enter degrees: " );
   gets ( szInput );
   n = atof ( szInput );
   m = sin (n*pi/180);
   printf ( "The sine of %f degrees is %f\n" , n, m );
   return 0;
}
```

# Преобразование типов

**double strtod( const char *nptr, char **endptr );**

- Convert strings to a double-precision value.

**Nptr -** Null-terminated string to convert;

**endptr** - Pointer to character that stops scan.

**Return Value**

strtod returns the value of the floating-point number, except when the representation would cause an overflow, in which case the function returns +/–HUGE_VAL. The sign of HUGE_VAL matches the sign of the value that cannot be represented. strtod returns 0 if no conversion can be performed or an underflow occurs.

# Преобразование типов

```
#include <stdio.h>
 #include <stdlib.h>

int main ()
{
  char szOrbits[] = "365.24 29.53";
  char * pEnd;
  double d1, d2;

  d1 = strtod (szOrbits,&pEnd);
  d2 = strtod (pEnd,NULL);

  printf ("The moon completes %.2lf orbits per Earth year.\n", d1/d2);
  return 0;
}
```

# Преобразование типов

- **int atoi( const char *str )**

  **-** Convert a string to integer

- **long atol( const char *str )**

  - Convert a string to a long integer

- **long strtol( const char *nptr, char **endptr, int base )**

  - Convert strings to a long-integer value

- **unsigned long strtoul( const char *nptr, char **endptr, int base )**

  - Convert strings to an unsigned long-integer value.

# Генерация псевдослучайных последовательностей

**int rand( void )**

**-** Generates a pseudorandom number

**Return Value**

rand returns a pseudorandom number. The rand function returns a pseudorandom integer in the range 0 to RAND_MAX (32767).

---

**void srand( unsigned int seed )**

**-** Sets a random starting point.

# Генерация псевдослучайных последовательностей

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main ()
{
 printf ("First number: %d\n", rand() % 100);
 srand ( time(NULL) );
 printf ("Random number: %d\n", rand() % 100);
 printf ("Again the first number: %d\n", rand() %100);
 return 0;
 }
```

# Контроль процесса выполнения программы

**void exit ( int status )**

- Terminate calling process

**Status** - Status value returned to the parent process. Generally, a return value of 0 or EXIT_SUCCESS indicates success, and any other value or the constant EXIT_FAILURE is used to indicate an error or some kind of abnormal program termination.

```
int main ()
{
 FILE * pFile;
 pFile = fopen ("myfile.txt","r");
 if (pFile==NULL)
 {
   printf ("Error opening file");
   exit (1);
 }
 else
 { /* file operations here */ }
 return 0;
}
```

# Контроль процесса выполнения программы

**int atexit( void (__cdecl *func )( void ) )**

- Processes the specified function at exit.

```
void fnExit1 (void)
{
 puts ("Exit function 1.");
}

void fnExit2 (void)
{
 puts ("Exit function 2.");
}

int main ()
{
 atexit (fnExit1);
 atexit (fnExit2);
 puts ("Main function.");
 return 0;
 }
```

# Математика

**div_t div( int numer, int denom )**

- Computes the quotient and the remainder of two integer values

**Return Value**

```
struct div_t
{
        int quot;
        int rem;
}
```

```
int main ()
{
 div_t divresult;
 divresult = div (38,5);
 printf ("38 div 5 => %d, remainder %d.\n", divresult.quot, divresult.rem);
 return 0;
 }
```

# Сортировка

```
void qsort( void *base,
            size_t num,
            size_t width,
            int (__cdecl *compare )(const void *, const void *)
          );
```

- Performs a quick sort

**base** - Start of target array.

**Num** - Array size in elements.

**width** - Element size in bytes.

**compare** - Comparison function. The first parameter is a pointer to the key for the search and the second parameter is a pointer to the array element to be compared with the key.

# Сортировка

```
int values[] = { 40, 10, 100, 90, 20, 25 };

int compare (const void * a, const void * b)
{
       return ( *(int*)a - *(int*)b );
}

int main ()
{
 int n;
 qsort (values, 6, sizeof(int), compare);
 for (n=0; n<6; n++)
      printf ("%d ",values[n]);
 return 0;
}
```

# Поиск

```
void *bsearch ( const void *key,
                const void *base,
                size_t num,
                size_t width,
                 int ( __cdecl *compare ) ( const void *, const void *)
    );
```

**-** Performs a binary search of a sorted array.

**key** - Object to search for.

**base** - Pointer to base of search data.

**Num** **-** Number of elements.

**Width** **-** Width of elements.

**compare** - Callback function that compares two elements. The first is a pointer to the key for the search and the second is a pointer to the array element to be compared with the key.

**Return Value**

bsearch returns a pointer to an occurrence of key in the array pointed to by base. If key is not found, the function returns NULL. If the array is not in ascending sort order or contains duplicate records with identical keys, the result is unpredictable.

# Поиск

```
int compareints (const void * a, const void * b)
{
      return ( *(int*)a - *(int*)b );
}

int values[] = { 10, 20, 25, 40, 90, 100 };

int main ()
{
       int * pItem;
       int key = 40;
       pItem = (int*) bsearch (&key, values, 6, sizeof (int), compareints);
       if (pItem!=NULL)
                printf ("%d is in the array.\n",*pItem);
       else
                 printf ("%d is not in the array.\n",key);
      return 0;
 }
```

# Практика

1. В файле в каждой строке записано 20 дат в формате dd.mm.yyyy. Необходимо прочитать файл, отсортировать даты в хронологическом порядке и вывести на экран.
2. Написать функцию, возвращающую произвольное число на отрезке [a;b]