ФУНКЦИИ СТАНДАРТНОГО ВВОДА / ВЫВОДА

<stdio.h>

Функции посимвольного ввода / вывода

`int getchar()` – считывание одного символа из стандартного потока ввода ***stdin***

`int putchar(int c)` – вывод одного символа в стандартный поток вывода ***stdout;***
`int c` – код выводимого символа

Форматный ввод-вывод

```
int printf(char* format, arg1, arg2, ... );
```

Функция **printf()** возвращает количество напечатанных символов. Форматная строка содержит два вида объектов: обычные символы, которые напрямую копируются в выходной поток, и спецификации преобразования, начинающиеся знаком **%**.

Например:

```
printf("x=%d; y=%d", x, y);    →    x=7; y=15
```

Основные спецификации преобразования printf

Символы форматирования	Тип аргумента; вид печати
%d	**int** ; десятичное ***(decimal)*** целое
%o	**unsigned int** ; беззнаковое восьмеричное ***(octal)*** целое (без символа **0** слева)
%x , %X	**unsigned int** ; беззнаковое шестнадцатеричное ***(hexadecimal)*** целое (без символов **0x** или **0X** слева). Для **10 ... 15** используются **abcdef** или **ABCDEF**

Символы форматирования	Тип аргумента; вид печати
`%u`	`unsigned int ;` беззнаковое десятичное целое
`%c`	`int ;` одиночный символ
`%s`	`char* ;` строка – выводятся все символы строки, идущие до завершающего нуль-символа `'\0'`
`%f`	`float ; [-]m.dddddd`, где количество цифр дробных разрядов **d** задается точностью (по умолчанию равно 6)

Символы форматирования	Тип аргумента; вид печати
%e , %E	double ; [-]m.ddddde±xx или [-]m.ddddddE±xx , где количество цифр дробных разрядов d задается точностью (по умолчанию равно 6)
%g , %G	double ; использует %e или %E, если порядок по модулю больше, чем 4; в противном случае использует %f. Завершающие нули и завершающая десятичная точка не печатаются
%p	void* ; указатель (представление зависит от реализации)

Символы форматирования	Тип аргумента; вид печати
%%	аргумент не преобразуется; печатается знак %
%l	long ; например, %ld
%h	short ; например, %hd
%число	минимальная ширина поля форматирования
%-	аргумент выравнивается по левому краю поля

```
int scanf(char* format, p_arg1, p_arg2, ... );
```

Функция **scanf()** возвращает количество успешно введенных элементов данных. Аргументы ввода должны быть указателями.
Например:

```
int x, y;
scanf("%d%d", &x, &y);
```

ОПЕРАЦИИ СО СТРОКАМИ И СИМВОЛАМИ

<string.h>

Дозапись строки **t** в конец строки **s** :

```
strcat(char* s, char* t)
```

Дозапись **n** символов из строки **t** в конец строки **s** :

```
strncat(char* s, char* t, int n)
```

Сравнение двух строк. Возвращается отрицательное число, нуль или положительное число для **s < t** , **s == t** или **s > t** , соответственно :

```
strcmp(char* s, char* t)
```

Делает то же, что и **strcmp()** , но количество сравниваемых символов не может превышать **n** :

```
strncmp(char* s, char* t, int n)
```

Копирование строки **t** в строку **s** :

```
strcpy(char* s, char* t)
```

Копирование не более **n** символов из строки **t** в строку **s** :

```
strncpy(char* s, char* t, int n)
```

Возвращает длину строки **s** :

```
strlen(char* s)
```

Возвращает указатель на первое появление символа **c** в строке **s** или **NULL**, если символа **c** нет в строке **s** :

```
strchr(char* s, int c)
```

Возвращает указатель на последнее появление символа **c** в строке **s** или **NULL**, если символа **c** нет в строке **s** :

```
strrchr(char* s, int c)
```

<ctype.h>

Проверка, является ли символ **c** буквой

```
isalpha(int c)
```

Проверка, является ли символ **c** буквой верхнего регистра

```
isupper(int c)
```

Проверка, является ли символ **c** буквой нижнего регистра

```
islower(int c)
```

Проверка, является ли символ **c** цифрой

```
isdigit(int c)
```

Проверка, является ли символ **c** буквой или цифрой

```
isalnum(int c)
(isalpha(c) || isdigit(c))
```

Проверка, относится ли символ **c** к набору пробельных символов, таким, например, как собственно пробел `' '`, горизонтальная табуляция `'\t'`, новая строка `'\n'`, возврат каретки `'\r'`, перевод страницы `'\f'`, вертикальная табуляция `'\v'` и др.

```
isspace(int c)
```

Приведение символа **c** к верхнему регистру

```
toupper(int c)
```

Приведение символа **c** к нижнему регистру

```
tolower(int c)
```

<stdio.h>

Форматный вывод в строку **string** :

```
int sprintf(char* string, char* format,
            arg1, arg2, ... );
```

Форматный ввод из строки **string** :

```
int sscanf(char* string, char* format,
           p_arg1, p_arg2, ... );
```

<stdlib.h>

Преобразование содержимого строки **string** в число :

```
double atof(char* string)

int atoi(char* string)

long atol(char* string)
```

Исполнение команд операционной системы

Функция

```
int system(char* s)
```

выполняет команду системы, содержащуюся в строке **s** , и затем возвращается к выполнению текущей программы. Содержимое строки **s** зависит от особенностей конкретной операционной системы.