Б5. Помощни функции: <stdlib.h>

В заглавния файл <stdlib.h> са декларирани функции, извършващи преобразуване на числа, заделяне на памет и други подобни задачи.

```
double atof(const char *s)
```
atof преобразува s в double; функцията е еквивалентна на strtod(s, (char**)NULL).

```
int atoi(const char *s)
```
преобразува s в int; функцията е еквивалентна на (int)strtol (s, (char **)NULL, 10).

```
long atol (const char *s)
```
преобразува s в long; функцията е еквивалентна на strtol (s, (char**) NULL, 10).

```
double strtod(const char *s, char **endp)
```
strtod преобразува представката на s в double, като пренебрегва началните празни пространства; функцията запазва в *endp указател към наставката, която не е преобразувана (ако съществува такава), освен ако endp не е NULL. Ако резултатът надвиши зададените граници, върнатата стойност е HUGE_VAL със съответния знак; ако резултатът губи значност, върнатата стойност е нула. И в двата случая, стойността на errno е ERANGE.

```
long strtol (const char *s, char **endp, int base)
```
strtol преобразува представката на s в long, като пренебрегва началните празни пространства; функцията запазва в *endp указател към наставката, която не е преобразувана, освен ако endp не е NULL. Ако base има стойност между 2 и 36, преобразуването се извършва, като се приема, че входът е прочетен при такава основа. Ако base е нула, за основа се приема 8,10 или 16; ако в началото има 0, резултатът се приема за осмично число; Ox или Ox в началото показват шестнадесетично число. И в двата случая буквите представят цифрите от 10 до base-1; при основа 16 са разрешени записите Ox или OX. Ако резултатът надвиши зададените граници, върнатата стойност е LONG_MAX или LONG_MIN, в зависимост от знака на резултата; и в двата случая errno има стойност ERANGE.

```
unsigned long strtoul (const char *s, char **endp, int base)
```
strtoul е еквивалентна на strtol, с изключение на факта, че резултатът е unsigned long и стойността за грешка е ULONG_MAX.

```
int rand(void)
```
rand връща псевдослучайно цяло число със стойност между 0 и RAND_MAX; минималната стойност на RAND_MAX е 32767.

```
void srand(unsigned int seed)
```
srand използва seed като изходна стойност при генерирането на нова поредица от псевдослучайни числа. Първоначалната стойност на seed за генериране на псевдослучайни числа е 1.

```
void *calloc(size_t nobj, size_t size)
```
calloc връща указател към място в паметта за масив от nobj на брой обекта, всеки от които е с големина size; ако не може да открие такова място, calloc връща NULL. Мястото в паметта не е инициализирано.

```
void *malloc(size_t size)
```
malloc връща указател към място в паметта за обект с големина size; ако не може да открие такова място, malloc връща null. Мястото в паметта не е инциализирано.

```
void *realloc(void *p, size_t size)
```
realloc променя на size големината на обекта, към който сочи p. Съдържанието, което се е побирало в минималната от двете големини (старата и новата), остава непроменено. Ако новата големина надвишава от старата, новото заделено място в паметта не се инициализира, realloc връща указател към новото място в паметта. Ако заявката не може да се изпълни, realloc връща null, а *p остава непроменен.

```
void free(void *p)
```
free освобождава мястото в паметта, към което сочи p; ако p е null, функцията не извършва нищо. p трябва да бъде указател към място в паметта, което на по-ранен етап на програмата е било заделено със calloc, malloc или realloc.

```
void abort(void)
```
abort кара програмата да спре преждевременно изпълнението си, сякаш е използвана функцията raise (sigabrt).

```
void exit (int status)
```
exit предизвиква нормално прекратяване на програмата. Функциите atexit се извикват в обратен на регистрацията ред, отворените файлове изпразват буферите си, отворените потоци се затварят и контролът се предава на средата. Как точно status се връща на средата зависи от реализацията, но навсякъде status 0 се приема за успешно прекратяване на програмата. Могат също така да се използват стойностите EXIT_SUCCESS и EXIT_FAILURE.

```
int atexit(void (*fcn)(void))
```
atexit регистрира функцията fen, която да бъде извикана при нормално прекратяване на програмата; ако регистрацията не може да бъде изпълнена, функцията връща ненулева стойност.

```
int system(const char *s)
```
system подава низа s за изпълнение в средата. Ако s е null, system връща ненулева стойност, в случай че притежава програма, която може да се изпълни от командния интерпретатор на операционната система. Ако s не е null, върнатата стойност зависи от реализацията.

```
char *getenv(const char *name)
```
getenv връща низа от обкръжението на операционната система, който се асоциира с name, или null, ако такъв низ не съществува. Подробностите зависят от реализацията.

```
void *bsearch(const void *key, const void *base,
size_t n, size_t size,
int (*cmp)(const void *keyval, const void *datum))
```
bsearch претърсва елементите base [0] …base [n-1], като търси сред тях елемент, който съответства на *кеу. Функцията стр връща отрицателна стойност, ако първият й арумент (ключовата дума за търсене) има по-малка стойност от втория (обект от таблица); нула, ако двата аргумента са еднакви или положителна стойност, ако първият аргумент е по-голям от втория. Елементите в масива base трябва да бъдат подредени във възходящ ред. bsearch връща указател към съответстващия елемент или NULL, ако такъв елемент не съществува.

```
void qsort(void *base, size_t n, size_t size,
int (*cmp)(const void *, const void *))
```
qsort сортира във възходящ ред обектите от масива base [0] . . .base [n-1], като всеки обект е с големина size. Функцията за сравнение стр е еднаква с тази в bsearch.

```
int abs(int n)
```
abs връща абсолютната стойност на своя аргумент от тип int.

```
long labs(long n)
```
labs връща абсолютната стойност на своя аргумент от тип long.

```
div_t div(int num, int denom)
```
div изчислява частното и остатъка от делението num/denom. Резултатите се съхраняват в int членовете quot и rem на структура от тип div_t.

```
ldiv_t ldiv(long num, long denom)
```
ldiv изчислява частното и остатъка от делението num/denom. Резултатите се съхраняват в long членовете quot и rem на структура от тип ldiv_t.