Izvadak iz ASCII tablice

Znak	*Opis*	*Dekadska vrijednost*
LF	novi red (*new line*)	***10***
Space	praznina (*space, blank*)	***32***
0	znamenka nula	***48***
A	veliko slovo A	***65***
a	malo slovo a	***97***

Prikaz realnog broja

IEEE 754 - 32 bita	*IEEE 754 - 64 bita*
K = BE + 127	K = BE + 1023
denormaliziran: K = 0	denormaliziran: K = 0
± ∞ ili NaN: K = 255	± ∞ili NaN: K = 2047
najveći normaliziran $\approx 3.4 \times 10^{38}$	najveći normaliziran $\approx 1.8 \times 10^{308}$
najmanji pozitivan normaliz. $\approx 1.17 \times 10^{-38}$	najmanji pozitivan normaliz. $\approx 2.2 \times 10^{-308}$
$\lvert\rho\rvert \leq 2^{-24} \approx 6 \times 10^{-8}$	$\lvert\rho\rvert \leq 2^{-53} \approx 1.1 \times 10^{-16}$

Prefiksi za cjelobrojne konstante

`0`	: *oktalno*
`0x`, `0X`	: *heksadekadski*

Sufiksi za cjelobrojne konstante

`L`, `l`	: *long*
`LL`, `ll`	: *long long*
`U`, `u`	: *unsigned*

Sufiksi za realne konstante

`F`, `f`	: *float*
`L`, `l`	: *long double*

Standardna biblioteka

stdlib.h

```
size_t
EXIT_FAILURE, EXIT_SUCCESS
RAND_MAX, NULL

int abs(int x);                          |x|
long labs(long x);
long long llabs(long long x);

void srand(unsigned int seed);
int rand(void);

void exit(int status);
void *malloc(size_t size);
```

math.h

```
double fabs(double x);                   |x|
float fabsf(float x);
long double fabsl(long double x);

double pow(double x, double y);          x^y
double sqrt(double x);                   √x
double exp(double x);                    e^x
double log(double x);                    ln x
double log10(double x);                  log10 x

double sin(double x);
double cos(double x);
double tan(double x);
```

Prioritet i asocijativnost operatora

	Operator	Asocijativnost operatora
←Viši prioritet	*poziv funkcije* **()** **[] . ->** *postfiks* **++ --**	L → D
	! ~ sizeof *adresa* **&** *indirekcija* ***** *prefiks* **++ --** *unarni* **+ -**	D → L
	(cast)	D → L
	aritmetički *** / %**	L → D
	binarni **+ -**	L → D
	<< >>	L → D
	< <= > >=	L → D
Niži prioritet→	**== !=**	L → D
	bitovni **&**	L → D
	^	L → D
	|	L → D
	&&	L → D
	||	L → D
	? :	D → L
	= *= /= %= += -= **&= ^= |= <<= >>=**	D → L
	operator **,**	L → D

Zauzeće memorije (gcc, x86_64)

Tip	Broj bajtova
char	1
short	2
int	4
long	4
long long	8
float	4
double	8
long double	12

gcc opcije

-E
-std=c11
-Wall
-pedantic-errors
-S
-c
-o
-save-temps
--verbose

```
void *realloc(void *ptr, size_t size);
void free(void *ptr);
```

```
double asin(double x);
double acos(double x);
double atan(double x);
double atan2(double y, double x);

double cosh(double x);
double sinh(double x);
double tanh(double x);

double ceil(double x);                   ⌈x⌉
double floor(double x);                  ⌊x⌋

double fmod(double x, double y);
```

time.h

```
size_t, time_t
NULL
time_t time(time_t *timer);
```

string.h

```
size_t
NULL
char *strcpy(char *s1, const char *s2);
char *strncpy(char *s1, const char *s2, size_t n);
char *strcat(char *s1, const char *s2);
char *strncat(char *s1, const char *s2, size_t n);
size_t *strlen(const char *s);
int *strcmp(const char *s1, const char *s2);        s1 < s2 ⇒ strcmp < 0
int *strncmp(const char *s1, const char *s2, size_t n);
char *strchr(const char *s, int c);                 NULL ako nema
char *strrchr(const char *s, int c);                NULL ako nema
char *strstr(const char *s1, const char *s2);       NULL ako nema
char *strpbrk(const char *s1, const char *s2);      NULL ako nema
```

ctype.h

```
int isdigit(int c);       int iscntrl(int c);
int isdigit(int c);       int isspace(int c);
int isxdigit(int c);      int islower(int c);
int isalpha(int c);       int isupper(int c);
int isalnum(int c);       int toupper(int c);
int isprint(int c);       int tolower(int c);
```

stdio.h

```
stdin, stdout, stderr
size_t, FILE
NULL, EOF, SEEK_CUR, SEEK_END, SEEK_SET
```

```
int scanf(const char *format, ...);
int sscanf(const char *buffer, const char *format, ...);
int fscanf(FILE *stream, const char *format, ...);
```

opći oblik konverzijske specifikacije: `%[*][širina][modifikator]specifikator`

specifikatori za scanf: c, d, u, o, x, f, s, p, [znakovi], [^znakovi]

modifikatori za d, u, o, x : h - short l - long ll - long long

modifikatori za f : l - double L - long double

```
int printf(const char *format, ...);
int sprintf(char *buffer, const char *format, ...);
int fprintf(FILE *stream, const char *format, ...);
```

opći oblik konverzijske specifikacije: `%[znak][širina][.preciznost][modifikator]specifikator`

specifikatori za printf: c, d, u, o, x, X, f, e, E, g, G, s, p

modifikatori za d, u, o, x : h - short l - long ll - long long

modifikatori za f : l - double L - long double

```
int getc(FILE *stream);                int getchar(void);        greška, eof → EOF
int ungetc(int c, FILE *stream);                                 greška → EOF
int putc(int c, FILE *stream);         int putchar(int c);       greška → EOF

char *fgets(char *s, int n, FILE *stream);                       greška, eof → NULL
int fputs(const char *s, FILE *stream);    int puts(const char *s);greška → EOF

FILE *fopen(const char *filename, const char *mode);             greška → NULL
```

mode: "w[b]","a[b]","r[b]","w+[b]","a+[b]","r+[b]"

```
int fflush(FILE *stream);                                        greška → EOF
int fclose(FILE *stream);                                        greška → EOF

size_t fread(void *ptr, size_t size, size_t n, FILE *stream);        greška → <n
size_t fwrite(const void *ptr, size_t size, size_t n, FILE *stream); greška → <n

int fseek(FILE *stream, long offset, int whence);                greška → int≠0
```

whence:	SEEK_SET	u odnosu na početak datoteke
	SEEK_CUR	u odnosu na trenutačnu poziciju
	SEEK_END	u odnosu na kraj datoteke

```
long ftell(FILE *stream);                                        greška → -1L
```

Ključne riječi

auto
break
case
char
const
continue
default
do
double
else
enum
extern
float
for
goto
if
inline
int
long
register
restrict
return
short
signed
sizeof
static
struct
switch
typedef
union
unsigned
void
volatile
while
_Alignas
_Alignof
_Atomic
_Bool
_Complex
_Generic
_Imaginary
_Noreturn
_Static_assert
_Thread_local