Laboratorul 2. Tipuri de date. Operatori. Măsurarea timpului de execuție. Funcții matematice.

Tipuri fundamentale de date

Tip de date	Descriere
char	Număr pe **8 biți (1 byte)**.
int	Număr întreg a cărui **lungime** este **dependentă** de compilator și de sistemul de operare (ex. pe Linux, int se reprezintă pe 32 de biți și are gama de valori [-2^{31}, 2^{31} - 1]).
float	Număr real stocat în **virgulă mobilă**, în gama de valori 1.7E+/-38.
double	Număr real stocat în **virgulă mobilă**, în gama de valori 1.7E+/-308.

Specificatori

Pot fi adăugați tipurilor fundamentale de date:

Specificator	Descriere
short	Aplicabil doar pentru **int**. Tipul rezultat are cel puțin **16 biți**.
long	Aplicabil pentru **int** și **double**. **long int** are **cel puțin 32 biți**, iar **long double** are o **dimensiune mai mare decât double**.
signed	Aplicabil pentru **int** și **char**. O variabilă declarată **int** este **implicit signed**. O variabilă de tip **signed char** va putea lua orice valoare din intervalul [**-2^7, 2^7 - 1**].
unsigned	Aplicabil doar **tipurilor întregi** precizând faptul că **valoarea** variabilei este **pozitivă**.

Tipuri cu dimensiune specificată exact

Definite în header-ul **<stdint.h>** (ex: uint32_t).

Overflow

Valoarea conținută de o variabilă **depășește limitele** impuse de tipul de date folosit.

Legendă tabel operatori

P = Precedență, As = Asociativitate
->= stânga-dreapta, <- = dreapta-stânga

Operatori

P	Operator	Descriere	As
1	[ ]	Indexare	->
	. şi ->	Selecţie membru (prin structură, respectiv pointer)	->
	++ şi --	Postincrementare și postdecrementare	->
2	!	Negare logică	<-
	\	Complement faţă de 1 pe biţi	<-
	++ şi --	Preincrementare și predecrementare	<-
	+ şi -	+ şi - unari	<-
	*	Dereferenţiere	<-
	&	Operator adresă	<-
	(tip)	Conversie de tip	<-
	sizeof()	Mărimea în octeţi	<-
3	*	Înmulţire	->
	/	Împărţire	->
	%	Restul împărţirii	->
4	+ şi -	Adunare/scădere	->
5	<<și >>	Deplasare stânga/dreapta a biţilor	->
6	<	Mai mic	->
	≤	Mai mic sau egal	->
	>	Mai mare	->
	≥	Mai mare sau egal	->
7	==	Egal	->
	!=	Diferit	->
8	&	ŞI pe biţi	->
9	∧	SAU-EXCLUSIV pe biţi	->
10	|	SAU pe biţi	->
11	&&	ŞI logic	->
12	||	SAU logic	->
13	?:	Operator condiţional	<-
14	=	Atribuire	<-
	+= şi -=	Atribuire cu adunare/scădere	<-
	*= şi /=	Atribuire cu multiplicare/împărţire	<-
	%=	Atribuire cu modulo	<-
	&= și |=	Atribuire cu ŞI/SAU	<-
	∧=	Atribuire cu SAU-EXCLUSIV	<-
	<<= şi >>=	Atribuire cu deplasare de biţi	<-
15	,	Operator secvenţă	->

Funcții matematice

Antet	Descriere
double floor(double x)	Partea întreagă inferioară
double ceil(double x)	Partea întreagă superioară
double pow(double base, double exp)	$base^{exp}$
double sqrt(double x)	Rădăcina pătrată
double fabs(double x)	Valoarea absolută

Măsurarea timpului de execuție

Funcția **clock()** întoarce o aproximare a **numărului de cicluri de ceas** trecute **de la pornirea programului**. Pentru a obține numărul de secunde, se împarte valoarea la constanta **CLOCKS_PER_SEC**.

```
#include <stdio.h>
#include <time.h>
// Marcăm momentul de început
clock_t t_start = clock();
// Executăm operația pentru care măsurăm
    timpul de execuție [....]
// Marcăm momentul de sfârșit
clock_t t_stop = clock();
float seconds = ((float)(t_stop - t_start)
    ) / CLOCKS_PER_SEC;
```

Generarea numerelor aleatoare

- Funcția **rand()** întoarce o valoare cuprinsă între 0 și o valoare maximă dependentă de librărie.
- Numerele generate sunt **pseudo-aleatoare** și **dependente de** prima valoare, numită **seed**, setată prin intermediul funcției:

```
void srand(unsigned int seed)
```

- Cel mai des, seed-ul este inițializat cu **valoarea ceasului sistemului** de la pornirea programului:

```
srand((unsigned)time(NULL));
d = rand(); // generează valori random.
```

- Funcția **time()** întoarce **numărul de secunde trecute de la ora 00:00, din data de 1 ianuarie 1970** și primește ca parametru adresa unei variabile în care se salvează valoarea returnată.