1. **TIPURI DE DATE. CITIREA ŞI TIPĂRIREA DATELOR**
	1. **SCOPUL LUCRĂRII**

Lucrarea are ca scop familiarizarea studentului cu principalele tipuri simple de date definite în limbajul C++ şi cu citirea acestora de la tastatură şi tipărirea lor pe monitor.

De asemenea, se exersează citirea şi tipărirea datelor de tip character şi şir de caractere.

 Sunt utilizate instrucţiunile, comenzile, funcţiile şi macrourile

* getch ( )
* putch ( )
* getch ( )
* getchar ( )
* putchar ( )
* gets ( )
* puts ( )
* scanf ( )
* printf ( )

Se exersează formatarea şi alinierea datelor tipărite pe monitor

* 1. **BREVIAR TEORETIC**

În limbajul C++ sunt utilizate următoarele tipuri simple de date numerice:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Tipul de dată** | **Lungime****(octeţi)** | **Domeniu de valori** |
| **Întreg** | *signed char* | 1 | -128 ... 127 (-27 … 27-1) |
| *[unsigned] char* | 1 | 0 ... 255 (0 ... 28-1) |
| *int* | 2 | -32768 ... 32767 (-215 … 215-1) |
| *unsigned [int]* | 2 | 0 ... 65535 (0 ... 216-1) |
| *long [int]**unsigned long [int]* | 44 | -231 ... 231-1 (-231 … 231-1)0 ... 232-1 (0 ... 232-1) |
| **Real** | *float* | 6 | 3.4\*10-38 ... 3.4\*1038 |
| *double* | 8 | 1.7\*10-308 ... 1.7\*10308 |
| *long double* | 10 | 3.4\*10-4932 ... 3.4\*104932 |

Cel mai adesea, pentru citirea/scrierea datelor se utilizează cin>>/cout<<, dar formatarea este în acest caz în seama utilizatorului; ca alternativă, se pot utiliza funcţiile scanf/printf din limbajul C.

Funcţia ***getch( )*** citeste fără ecou pe monitor un caracter de la tastatură

Funcţia ***putch( )*** tipăreşte un caracter pe ecran

Funcţia ***getche( )*** citeste cu ecou pe monitor un caracter de la tastatură

Macro-ul ***getchar( )*** citeşte un caracter tastat, după asarea tastei ENTER

Macro-ul ***putchar( )*** afişează un caracter pe monitor

Macro-urile getchar si putchar(x) au prototipurile in header-ul <stdio.h>

Funcţia ***gets( )*** citeşte un şir de caractere

Funcţia ***puts( )*** afişează un şir de caractere

**FUNCŢIA scanf( )**

Funcţia de citire cu format ***scanf( )*** are sintaxa :

scanf(“lista de formate” , adresa\_var1 , adresa\_var2,…..);

- citeşte din fişierul standard de intrare ***stdio*** o secvenţă de câmpuri de intrare,caracter cu caracter, până la terminarea introducerii câmpurilor şi apăsarea tastei < Enter > ;

- formatează fiecare câmp conform formatului specificat în lista de formate.

Dincaracterele citite se calculează valori numerice sau literale, conform tipului fiecăreivariabile, dimensiunilor de format specificate şi a separatorilor de câmpuripredefiniţi (spaţiu, tab şi enter) sau impuşi explicit ;

- valorile astfel construite sunt stocate la adresele variabilelor specificate ca argumente; ordinea formatelor variabilelor trebuie să coincidă cu ordinea listei adreselor variabilelor în care se face citirea. Fiecare variabilă care se doreşte a fi citită trebuie corelată cu un format specific.

***Observaţie***: indiferent de formatul folosit, la întâlnirea unui spaţiu în introducereadatelor, este terminată citirea variabilei. Pentru funcţia de citire scanf trebuie folosit operatorul adresă “***&***”

Pentruvariabilele citite cu această funcţie trebuie precizate adresele la care se stochează înmemoria calculatorului valorile variabilelor. Funcţia va introduce valorile citite direct laacele adrese. Singurul caz în care nu este obligatorie folosirea operatorul adresăpentru citirea valorii unei variabile cu funcţia scanf  este citirea unui şir de caractere.

***Observaţie***: citirea cu ajutorul funcţiei scanf  a şirurilor de caractere care conţin spaţii este imposibilă. În cazul în care formatul specificat este necorespunzător, rezultatul obţinut poate fi neprevăzut. Valoarea întoarsă de scanf   în caz de succes, este numărul de variabile care au fost citite corect. Dacă nu a fost citită nici o variabilă (de exemplu s-a introdus un şir în loc de un număr) funcţia întoarce valoarea 0. Dacă apare o eroare înaintea oricărei citiri şi asignări, funcţia returnează EOF (constantă de sistem avândvaloarea întreagă –1).

***Specificatorii de format ai funcţiei scanf( ) :***

%c Citeşte un caracter

%d Citeşte un întreg zecimal

%i Citeşte un întreg zecimal

%e Citeşte un număr float

%f Citeşte un număr float

%g Citeşte un număr float

%o Citeşte un număr octal fără semn

%s Citeşte un şir de caractere

%x Citeşte un număr hexazecimal fără semn

%p Citeşte un pointer

%n Argumentul asociat primeşte o valoare întregă egală cu numărulde caractere deja citite

%u Citeşte un număr întreg fără semn

În specificatorul de format pot să apară şi modificatori de tip.

* ***Modificatorul h***, care poate precede caracterele de conversie d, i , o , u , x sau X, precizează că valoareaconvertită trebuie memorată ca un „short int” sau „unsigned short int”.
* ***Modificatorul l***, poate precede caracterele de conversie d, i, o, u, x, X, caz în care valoarea trebuie memorată ca un „ long int” sau „unsigned long int”, sau poate precede caracterele e , E , f  , g , G , caz în care valoarea trebuie memorată ca un „ double”.
* ***Modificatorul L*** poate precede numai caracterele e, E, f , g, G şi precizează că valoarea convertită trebuie memorată ca un „long double”

**FUNCŢIA printf( )**

Este perechea funcţiei scanf( ), semnificaţia caracterelor de control fiind asemănătoare.

Sintaxa de utilizare este :

int printf(“mesaje si lista de formate”, expr\_1, expr\_2, ….,expr\_n);

Funcţia printf  realizează următoarele :

- acceptă o serie de argumente de tip expresie pe care, după ce le evaluează, letransformă în şiruri de caractere conform formatului specificat

- scrie şirurile în fişierul standard de ieşire (sunt acceptate secvenţele de evitare). Dacă numărul de argumente specificate în format nu corespunde cu numărulargumentelor din lista de expresii, atunci apar rezultate neaşteptate care pot aveaefecte dăunătoare. Rezultatul întors de funcţie, în caz de succes, este numărul deocteţi scrişi, iar în caz de eroare, valoarea întoarsă este EOF.

Specificatorii deformat folosiţi pentru printf  sunt

%e , %E Număr real de forma iiii.zzzzzz , unde nr.zecimale z este dat de precizie (6 implicit)

%f Număr real de forma i.zzzzzz  , unde nr. zecimale este dat deprecizie (6 implicit) şi pentru partea întreagă este folosită doar o cifră

%g , %G Număr real care suprimă caracterele terminale care nuinfluenţează valoarea , adică cifrele 0 de la sfârşit şipunctul zecimal , dacă are partea fracţionară 0

%i Număr întreg în baza 8, 10, sau 16 în funcţie de primul sauprimele două caractere

%d Număr întreg în baza 10

%o Număr întreg în baza 8 ; nu este necesară scrierea cifrei 0la începutul numărului

%x Număr întreg în baza 16 ; nu este necesară scriereasecvenţei 0x la începutul numărului

%u Număr întreg fără semn

%s Şir de caractere

%c Un singur character

Expresiile afişate se pot alinia la stânga sau la dreapta şi se poate forţa afişareasemnului astfel :

- semnul plus (+) afişează explicit semnul expresiei

- semnul minus (-) aliniază expresia afişată la stânga

- absenţa oricărui semn semnifică alinierea expresiei afişate la dreaptă

Pentru numerele întregi şi pentru şirurile de caractere se poate specifica un număr care înseamnă spaţiul folosit pentru afişare. Dacă spaţiul necesar este mai mic

sau egal cu numărul specificat, atunci se vor afişa suplimentar spaţii (sau zerouri,dacă numărul este precedat de cifra 0) până la completarea spaţiului de afişare.

Pentru numerele reale se pot specifica, opţional, semnul pentru aliniere şi două numere separate prin punct. Primul precizează dimensiunea totală de afişare, iar al doilea precizia, adică numărul de zecimale afişate.

În cazul şirurilor de caractere, specificarea a două numere separate prin punct indică faptul că primul număr reprezintă numărul de caractere din şir care se vor afişa, iar al doilea reprezintă limita superioară de tipărire, completarea făcându-se cu spaţii la dreapta sau stânga, în funcţie de modul de aliniere.

Poate apare fenomenul de trunchiere a şirului afişat în cazul în care dimensiunea acestuia depăşeşte limita inferioară.

În cazul unui număr întreg, al doilea număr indică o completare la stânga cu zerouri până se ajunge la dimensiunea de afişare specificată.

În format se pot utiliza şi modificarii de tip ***h*** şi ***l*** , corespunzători lui „short”, respectiv „long”.

Dacă h este urmat de un ***d*** , ***i*** , ***o*** , ***u*** , ***x*** sau ***X***, atunci aceasta este o specificare de conversie relativ la „short int” sau „unsigned short int”

Dacă ***l*** este urmat de ***d***, ***i***, ***o***, ***u***, ***x*** sau ***X***, atunci specificarea de conversie se aplică unui argument „long int” sau „unsigned long int”.

Modificatorul ***L*** poate fi urmat de ***e*** , ***E*** , ***f***  , ***g*** sau ***G*** şi atunci specificarea de conversie se plică unui argument „long double”

* 1. **EXEMPLE**
1. ***Funcţiile getch( ) şi putch( )***

*#include <conio.h>*

*// Functia getch() citeste fara ecou un caracter*

*// Caracterul tastat va aparea pe ecran doar datorita lui putch(x)*

*void main(void)*

*{*

*clrscr();*

*putch(getch());*

*getch();*

*}*

1. ***Funcţia getche( )***

*#include <conio.h>*

*// Functia getche() face acelasi lucru ca getch(), dar cu ecou pe monitor*

*// Caracterul tastat va aparea de 2 ori (odata datorita lui getche(),*

*// odata datorita lui putch(x) ).*

*void main(void)*

*{*

*clrscr();*

*putch(getche());*

*getch();*

*}*

1. ***Macrourile getchar( ) şi putchar( )***

*#include <conio.h>*

*#include <stdio.h>*

*// Si macro-ul getchar citeste un caracter tastat, dar, spre deosebire*

*// de functiile getch() si getche(), doar dupa apasarea tastei ENTER*

*// Macro-urile getchar si putchar(x) au prototipurile in header-ul stdio.h*

*// (STanDard Input-Output)*

*// conio.h este inclus pentru functia getch()*

*void main(void)*

*{*

*clrscr();*

*putchar(getchar());*

*getch();*

*}*

1. ***Secvenţele ESCAPE; funcţia puts( )***

*#include <conio.h>*

*#include <stdio.h>*

*// Functia puts(x) scrie pe monitor sirul x*

*// Ghilimelele si apostroful, avand in mod normal semnificatii speciale,*

*// se scriu precedate de un back-slash \ , daca vrem sa apara ca atare in text*

*// Acum, deoarece se vede ca si back-slashul are o semnificatie speciala,*

*// trebuie si el precedat de un back-slash, daca vrem sa apara in text*

*void main(void)*

*{*

*puts("Ce frumoase ghilimele (\") avem !");*

*puts("Si apostroful (\') este frumos !");*

*puts("nu trebuie uitat nici back-slash-ul (\\) !");*

*getch();*

*}*

1. ***Funcţia printf( ) pentru scrierea cu format***

*#include <stdio.h>*

*#include <conio.h>*

*// Cea mai complexa functie de tiparire - printf - are prototipul in stdio.h;*

*// permite afisarea datelor sub comanda unor formate introduse cu simbolul %*

*// Back-slashul isi pastreaza semnificatia de introducere a unor comenzi*

*// De ex. \n - trecerea la un rand nou*

*// \t - tabulare orizontala etc.*

*// In tot programul, introducerea se face cu funtia deja studiata getch()*

*void main(void)*

*{*

*clrscr();*

*int i=123;*

*printf("\n\nDe aici incepe. Nu se va apasa niciodata tasta Enter !\n");*

*printf("Tasteaza un caracter. Caracterul va fi afisat\n");*

*printf("%c\n\n",getch());*

*printf("Tasteaza un caracter. Caracterul va fi aliniat la dreapta\n");*

*printf("\*%4c\*\n\n",getch());*

*printf("Tasteaza un caracter. Caracterul va fi aliniat la stanga\n");*

*printf("\*%-4c\*\n\n",getch());*

*printf("Apasa orice tasta. Va fi afisat sirul abc\n");*

*printf("%s\n\n","abc",getch());*

*printf("Apasa orice tasta. Sirul abc va fi afisat aliniat la dreapta\n");*

*printf("\*%10s\*\n\n","abc",getch());*

*printf("Apasa orice tasta. Sirul abc va fi afisat aliniat la stanga\n");*

*printf("\*%-10s\*\n\n","abc",getch());*

*printf("Apasa orice tasta. Va fi afisata valoarea variabilei i (i=123).\n");*

*printf("%d\n\n",i,getch());*

*printf("Apasa orice tasta. Valoarea lui i va fi afisata aliniata la dreapta\n");*

*printf("\*%10d\*\n\n",i,getch());*

*printf("Apasa orice tasta. Valoarea lui i va fi afisata aliniata la stanga\n");*

*printf("\*%-10d\*\n\n",i,getch());*

*printf("Apasa orice tasta. Valoarea lui i va fi afisata cu zerouri in fata\n");*

*printf("\*%010d\*\n\n",i,getch());*

*printf("Pentru terminare, apasati orice tasta");*

*getch();*

*}*

1. ***Utilizarea funcţiilor printf( ) şi scanf ( )***

*#include <conio.h>*

*#include <stdio.h>*

*// Deosebirile intre introducerea datelor de tip sir cu functia gets(x)*

*// si introducerea cu ajutorul functiei scanf(x)*

*// Si tiparirea se va face cu doua functii diferite puts(x) si Printf(x)*

*void main(void)*

*{*

*char sir[50];*

*clrscr();*

*printf("\nSiruri citite cu functia ");*

*printf("gets()\n");*

*printf("\nIntroduceti un sir FORMAT DIN MAI MULTE CUVINTE\n");*

*gets(sir);*

*printf("\nSirul tiparit cu functia puts() :\n");*

*puts(sir);*

*printf("\nSirul tiparit cu functia printf(%%s) :\n");*

*printf("%s\n",sir);*

*printf("\nPentru continuare, apasati o tasta !");*

*getch();*

*clrscr();*

*printf("\nSiruri citite cu functia ");*

*printf("scanf(%%s)\n");*

*printf("\nIntroduceti un sir FORMAT DIN MAI MULTE CUVINTE\n");*

*scanf("%s",sir);*

*printf("\nSirul tiparit cu functia puts() :\n");*

*puts(sir);*

*printf("\nSirul tiparit cu functia printf(%%s) :\n");*

*printf("%s\n",sir);*

*printf("\nPentru terminare, apasati o tasta !");*

*getch();*

*}*

1. ***Exerciţii asupra tipurilor de date***

*#include <conio.h>*

*#include <stdio.h>*

*// Diferentele intre tipul intreg int (format de tiparire %d)*

*// si tipul real double (format %lf)*

*void main()*

*{*

*clrscr();*

*printf("Se afiseaza valoarea polinomului 3x^2-8x+7\n\*

*pentru o valoare intreaga a lui x, introdusa de la tastatura\n");*

*int x;*

*printf("tastati valoarea lui x=");*

*scanf("%d",&x);*

*printf("x=%d\tp(x)=%d\n\n",x,3\*x\*x-8\*x+7);*

*printf("\nPentru a continua programul, apasati orice tasta\n\n");*

*getch();*

*printf("Se afiseaza valoarea polinomului 3.5y^3-9.8y+3.7\n\*

*pentru o valoare flotanta dubla precizie a lui y,\n\*

*valoare introdusa de la tastatura\n");*

*double y;*

*printf("tastati valoarea lui y=");*

*scanf("%lf",&y);*

*printf("y=%lf\tp(z)=%lf\n\n",y,3.5\*y\*y\*y-9.8\*y+3.7);*

*printf("\nPentru a continua programul, apasati orice tasta\n\n");*

*getch();*

*printf("\nPrecizia maxima intr-un calcul este de 19-20 cifre semnificative\n");*

*printf("\n2/3=%20.18Lf",2/3.0l);*

*printf("\n\nPentru a incheia programul, apasati orice tasta\n");*

*getch();*

*}*

1. ***Formatele zecimal, octal şi hexagesimal***

*#include <conio.h>*

*#include <stdio.h>*

*// Limbajul C poate tipari rezultatele in zecimal (formatul %d),*

*// octal (formatul %o) sau hexagesimal (formatul %x)*

*void main()*

*{*

*clrscr();*

*int x;*

*printf("tastati valoarea lui x=");*

*scanf("%d",&x);*

*printf("Valoarea in zecimal este\n");*

*printf("x(10)=%d\n\n",x);*

*printf("Valoarea in octal este\n");*

*printf("x(8)=%o\n\n",x);*

*printf("Valoarea in hexagesimal este\n");*

*printf("x(16)=%X\n\n",x);*

*getch();*

*}*

* 1. **TEME DE LABORATOR**
1. Se citesc de la tastatură coordonatele (xa, ya) şi (xb, yb) (valori întregi) a două puncte A şi B din plan. Să si tipărească pe monitor lungimea segmentului AB.
2. Se citesc de la tastatură lungimile (valori reale) a, b şi c ale unui triunghi. Se va calcula cu ajutorul formulei lui Heron aria tringhiului şi se va afişa pe monitor, ca în exemplul următor:

date de intrare: a=3, b=4, c=5;

afişarea dorită: Aria triunghiului cu laturile de lungime 3, 4 şi 5 este 6;

1. Se citesc de la tastatură valorile întregi a şi b; se va tipari câtul şi restul împărţirii întregi a lui a la b, ca în exemplul următor:

date de intrare: a=14, b=3;

afişarea dorită: a:b = 4 si rest 2

1. Se citesc de la tastatură valorile întregi a şi b; se va tipari câtul şi restul împărţirii întregi a lui a la b, ca în exemplul următor:

date de intrare: a=14, b=3;

afişarea dorită: 14:3 = 4 si rest 2

1. Se citesc trei valori reale a, b, şi c.

Se calculează şi se afişează pe monitor valoarea polinomului

P(x) = a\*x2 + bx + c

 pentru patru valori ale parametrului x, citite de la tastatură, ca în exemplul următor:

 Presupunând că s-au introdus pentru a, b şi c valorile 1, 2 şi 3 (respectiv), iar pentru cele patru valori ale parametrului x valorile succesive 0, 1, 2 şi 3, pe monitor trebuie să apară mesajele

 P(0) = 3

 P(1) = 6

 P(2) = 11

 P(3) = 18