FUNCŢII DE INTRARE/IEŞIRE STANDARD

1. Conţinutul lucrării

În lucrare sunt prezentate funcţiile I/E standard, adică funcţiile din biblioteca compilatorului C/C++, care realizează citirea/scrierea din/în fişierele standard de I/E.

2. Consideraţii teoretice

Terminalul standard este terminalul de la care s-a lansat programul. Terminalului standard îi sunt ataşate două fişiere: de intrare (stdin) şi de ieşire (stdout). Ele sunt fişiere secvenţiale.

Funcţiile din biblioteca compilatorului C/C++ utilizate mai frecvent pentru operaţiile de I/E sunt:

- pentru intrare: **getch**, **getche**, **gets**, **scanf**, **sscanf** ;
- pentru ieşire: **putch**, **puts**, **printf**, **sprintf.**

2.1. Funcţiile getch, getche şi putch

Funcţia **getch** citeşte fără ecou un caracter prin apăsarea unei taste. Tasta poate avea un corespondent ASCII sau o funcţie specială. În primul caz funcţia returnează codul ASCII al caracterului. În al doilea caz, funcţia se apelează de două ori: prima dată returnează valoarea zero, iar a doua oară returnează o valoare specifică tastei acţionate.

Funcţia **getche** este analogă cu funcţia **getch**, realizând însă citirea cu ecou.

Apelul funcţiilor **getch** şi **getche** conduce la aşteptarea apăsării unei taste.

Funcţia **putch** afişează pe ecranul terminalului un caracter corespunzător codului ASCII transmis ca parametru. Caracterele imprimabile au codul ASCII în intervalul [32,126]. Pentru coduri în afara acestui interval se afişează diferite pictograme. Funcţia returnează valoarea parametrului de la apel.

Prototipurile acestor trei funcţii se găsesc în fişierul **conio.h** şi sunt:

```
int getch(void);
int getche(void);
int putch(int ch);
```

Exemplu de utilizare:

```
/* Programul  L2Ex1 */

#include <conio.h>
int main()
{
        putch(getch());
        return 0;
}
```

2.2. Funcţiile gets *şi* puts

Funcţia **gets** citeşte cu ecou de la terminalul standard un şir de caractere ale codului ASCII, la adresa specificată drept parametru al funcţiei. Din funcţie se revine la:

- citirea caracterului '\n' (newline), caracter care este transformat în caracterul '\0' (null). În acest caz funcţia returnează adresa de început a zonei de memorie în care se păstrează caracterele;
- citirea sfârşitului de fişier (CTRL/Z), funcţia returnând valoarea zero.

Funcţia **puts** afişează la terminalul standard un şir de caractere corespunzând codului ASCII de la adresa transmisă ca parametru. Caracterul '\0' este interpretat ca '\n'. Funcţia returnează codul ultimului caracter afişat sau –1 în caz de eroare.

Prototipurile funcţiilor se găsesc în fişierul **stdio.h** şi sunt:

```
char * gets (char *s);
int puts (const char *s);
```

Exemplu de utilizare:

```
/*   Programul L2Ex2  */

#include <stdio.h>
#include <conio.h>
int main()
{
        char s[200];
        printf("Introduceti un sir de caractere urmat de ENTER\n");
        gets(s);
        printf("Sirul de caractere introdus\n");
        puts(s);
        return 0;
}
```

2.3. Funcţiile scanf şi printf

Funcţia **scanf** are rolul de a introduce date tastate de la terminalul standard sub controlul unor formate. Datele introduse sunt convertite din formatele lor externe în formate interne şi sunt păstrate la adresele specificate la apel. Datele introduse se termină cu apăsarea tastei ENTER.

Prototipul funcţiei **scanf** se găseşte în fişierul **stdio.h** şi este:

```
int scanf(const char *format [,adresa,...]);
```

Ea returnează numărul de câmpuri de la intrare introduse corect sau valoarea EOF(-1) în cazul întâlnirii sfârşitului de fişier (CTRL/Z).

Formatul este specificat ca un şir de caractere. El conţine specificatorii de format, care definesc conversiile din formate externe în formate interne. Un specificator de format este alcătuit din:

- caracterul %;
- opţional caracterul *, care indică faptul că data prezentă la intrare nu se atribuie nici unei variabile;
- opţional un număr zecimal, care defineşte lungimea maximă a câmpului controlat de format;
- 1 sau 2 litere, care definesc tipul conversiei.

Câmpul controlat de format începe cu primul caracter curent care nu este alb şi se termină, după caz:

a) la caracterul după care urmează un caracter alb;
b) la caracterul care nu corespunde tipului de conversie;
c) la caracterul la care se ajunge la lungimea maximă a câmpului.

Datele se citesc efectiv după apăsarea tastei ENTER. Adresa unei variabile se specifică prin **&nume_variabilă.**

Literele care definesc tipul conversiei sunt:

Litera	**Tipul datei citite**
c	char
s	şir de caractere
d	întreg zecimal
o	întreg octal
x, X	întreg hexazecimal
u	unsigned
f	float
ld, lo, lx, lX	long
lu	unsigned long
lf/ Lf	double/long double

Funcţia **printf** este folosită pentru afişarea unor date pe ecranul terminalului standard sub controlul unor formate. Datele sunt convertite din format intern în formatul extern specificat.

Prototipul funcţiei **printf** se găseşte în fişierul **stdio.h** şi este:

int printf(const char *format [,expresie, …]);

Formatul este dat ca un şir de caractere. El are în structura sa succesiuni de caractere (care se afişează) şi specificatori de format.

Un specificator de format conţine:
- caracterul %;
- opţional caracterul minus -, care specifică cadrarea datei în stânga câmpului (implicit cadrarea se face în dreapta);
- opţional un număr zecimal, care defineşte dimensiunea minimă a câmpului în care se afişează data;
- opţional un punct urmat de un număr zecimal, care specifică precizia de afişare a datei;

- una sau două litere, care definesc tipul conversiei. Faţă de literele prezentate la scanf apar literele e şi E pentru afişarea datelor float sau double sub formă de exponent, g şi G pentru afişarea datelor sub forma de exponent sau nu, astfel ca data afişată să ocupe un număr minim de caractere.

Funcţia returnează numărul de caractere (octeţi) afişate la terminal sau –1 în caz de eroare.

Exemple de folosire:

```
/*  Programul L2Ex3  */

#include <stdio.h>
#include <conio.h>
int main( )
{
        int a;
        float b,c;
        printf("Introduceti o valoare intreaga a=");
        scanf("%5d",&a);
        printf("Introduceti o valoare reala b=");
        scanf("%5f",&b);
        c=a+b;
        printf("Valoarea c=a+b este: %6.3f\n",c);
        return 0;
}
```

2.4. Funcţiile sscanf şi sprintf

Faţă de funcţiile **scanf** şi **printf**, funcţiile **sscanf** şi **sprintf** au în plus ca prim parametru adresa unei zone de memorie care conţine caractere ASCII. Funcţia **sscanf** citeşte caracterele din această zonă de memorie în loc de zona tampon corespunzătoare fişierului standard de intrare (tastaturii). Funcţia **sprintf** depune caracterele în această zonă de memorie în loc de a fi afişate pe ecran.

Prototipurile acestor funcţii se găsesc în fişierul **stdio.h** şi sunt:

```
int scanf (const char *buffer, const char *format [,adresa, ..]);
int sprintf (char *buffer, const char *format [,adresa, …);
```

Exemplu de folosire:

```
/* Programul L2Ex4 */

#include <stdio.h>
#include <conio.h>
int main ( )
{
        char s[100], q[100];
        int a,b;
        float c,d;
```

```
        printf ("Introduceti in acelasi rand valoarea \
                lui a si b despartite intre ele prin spatiu \
                urmate de ENTER\n");
        gets(s);
        sscanf(s, "%d %f", &a, &c);
        printf("a=%4d  c=%8.3f\n",a,c);
        sprintf(q, "%4d  %8.3f\n",a,c);
        sscanf(q, "%d %f",&b,&d);
        printf("\n b=%5d  d=%9.4f\n",b,d);
        return 0;
}
```

3. Mersul lucrării

3.1. Se vor executa programele date ca exemplu în lucrare şi în curs şi se vor analiza rezultatele obţinute.

3.2. Scrieţi un program pentru a verifica modul de execuţie a funcţiei getch când se apasă o tastă specială (F1…F12, CTRL/tastă).

3.3. Scrieţi un program pentru a verifica ce se afişează de către funcţia putch atunci când parametrul său este o valoare în afara intervalului [32,126].

3.4. Scrieţi un program care afişează codul ASCII corespunzător unei taste apăsate.

3.5. Scrieţi un program care afişează caracterul corespunzător unui cod ASCII din intervalul [32,126].

3.6. Scrieţi un program care să conţină apelul gets(*s*), unde *s* a fost definit ca un tablou şir de caractere. Verificaţi ce conţine fiecare element al tabloului. De ce caracterul '\n' a fost înlocuit cu '\0'?

3.7. Scrieţi un program care citeşte o literă mică şi o afişează sub formă de literă mare.

3.8. Scrieţi un program care citeşte o literă mare şi o afişează sub formă de literă mică.

3.9. Scrieţi un program care realizează suma, diferenţa, produsul şi împărţirea a două numere reale. Afişarea se va face sub formă tabelară:

x	**y**	**x + y**	**x – y**	**x * y**	**x / y**

3.10. Scrieţi un program pentru a verifica modul de afişare a valorii lui $\pi = 3.14159265$ cu diferiţi descriptori de format.

3.11. Scrieţi un program pentru afişarea unui întreg zecimal citit de la tastatură în octal şi hexazecimal.