1. NOȚIUNI INTRODUCTIVE ALE LIMBAJULUI DE PROGRAMARE C

1.1. GENERALITĂȚI PRIVIND LIMBAJUL DE PROGRAMARE C

Limbajul C a fost creat inițial de Kernigan şi Ritchie pentru realizarea şi dezvoltarea sistemelor de operare UNIX. Ulterior limbajul C a devenit cel mai utilizat limbaj de programare atît pentru crearea şi dezvoltarea sistemelor de operare, cît şi pentru realizarea programelor de aplicație.

Lmbajul C este unicul limbaj de programare, care utilizează forma limbajelor de nivel înalt (BASIC, FORTRAN, PASCAL), permițînd în acelaşi timp acțiuni asupra hard-ului calculatorului, similare celor realizat prin intermediul limbajelor de asamblare. Numeroase operații efectuate pe un calculator personal cu ajutorul limbajului de asamblare pot fi realizate într-o manieră mult mai simplă cu ajutorul limbajului C. Cele mai bune compilatoare C (TURBO C, TURBO C++, BORLAND C) generează programe executabile a căror viteză de execuție doar în puține cazuri poate fi îmbunătățită prin rescrierea programului în limbajul de asamblare.

Fiind un limbaj bine structurat, prin sintaxa sa limbajul C uşurează activitatea de scriere a programelor, care sunt modulare şi uşor de înțeles. El oferă numeroase facilități destinate să ajute programatorul în realizarea şi depănare celor mai complexe programe de aplicație. Un alt avantaj oferit de limbajul C îl constitue protabilitatea programelor realizate cu ajutorul lui. Un program scris în C, destinat rulării pe un anumit tip de calculator, poate fi modificat cu uşurință în vedrea utilizării sale pe alte tipuri de calculatoare.

1.2. Mediul de programare C

Principalele activități legate de realizarea unui program de calcul (editarea fişierului sursă, compilarea, editarea legăturilor şi lansarea în execuție) necesită folosirea unor programe specializate (editoare de text, compilatoare, link-editoare). În sistemele de calcul actuale aceste programe sunt grupate într-un ansamblu unitar, de sine stătător, care poartă numele de **mediu de programare**.

Pentru limbajul C cele mai utilizate medii de programare sunt TURBO C (pentru elaborarea programelor C clasice) şi TURBO C++ sau Borland C (pentru elaborarea atît a programelor clasice cît şi a programelor C++ orientate pe obiect). Avînd în vedere faptul că limbajul C++ constituie un super-set al limbajului C clasic şi că mediul Turbo C++ permite atît dezvoltarea de programe C clasice, cît şi de programe C orientate pe obiect, în continuare ne vom referi numai la mediul Turbo C++ sau la varianta Borland C a acestuia, folosind notația Turbo C++/Borland C. Comunicarea programatorului cu mediul Turbo C++/Borland C se realizează prin intermediul comenzilor grupare într-un sistem de meniuri cu organizare ierarhică (un meniu principal din care derivă alte submeniuri).

1.3. Structura unui program C

În general, un program de calcul este o segvență de operații care se efectuează asupra unor date şi care implementează un algoritm sau o procedură de rezolvare a unei probleme. El constă dintr-o succesiune de caractere cu ajutorul cărora se alcătuiesc cuvintele şi propozițiile (instrucțiunile). Acestea trebuie să respecte setul de reguli ce definesc **sintaxa limbajului**. În limbajul C pentru scrierea textului ce constituie programul de calcul se pot folosi doar caracterele, ce alcătuiesc alfabetul acestuia. Alfabetul limbajului C cuprinde literele mari şi mici ale alfabetului latin, caracterul de subliniere "_" folosit ca element de legătură, cifrele zecimale de la 0 la 9 şi simbolurile speciale, care reprezintă operatorii, delimitatorii şi semnele de punctuație.

Tabelul 1

auto	else	register	union	cdecl
break	enum	return	unsigned	near
case	extern	short	void	_ds
char	float	signed	volative	far
const	for	sizeof	while	pascal
continue	goto	static	asm	_es
default	if	struct	_ss	huge
do	int	switch	interrupt	
double	long	typedef	_cs	

În limbajul C este prevăzut un set de **cuvinte cheie**, fiecare din ele avînd o semnificație specială (tabelul 1).

În numeroase situații practice procedure de calcul este lungă şi complicată, astfel încît programul devine deficil de imolementat, urmărit şi depănat. Tehnicile de programare procedurală oferă metode pentru înfățişarea şi structurarea programului în module uşor de realizat, înțeles şi întreținut. Aceste tehnici concentrează atenția asupra organizării secvenței de operatii în cadrul programului şi de cele mai multe ori ignoră structurile de date, asupra cărora se efectuiază operațiile. Spre deosebire de programarea procedurală, programarea orientată pe obiect mai întîi concentrează atenția asupra structurilor de date şi apoi asupra operațiilor, care se efectuează cu aceasta.

În limbajul C, care este un limbaj procedural , operațiile sunt grupate în blocuri de instrucțiuni delimitate de o pereche acolade "{ }". La rîndul lor, blocurile sunt în una sau mai multe funcții. O funcție C este un modul care grupează în interiorul unei perechi de acolade un set de operații codificate sub formă de instrucțiuni.

Datele asupra cărora funcția operează sunt fie întrările în funcție, fie date locale ale acesteia. La rîndul lor intrările în funcție pot fi date globale ale programului de calcul şi/sau date transmise funcției prin intermediul parametrilor acesteia. Rezultatele obținute în urma efectuării operațiilor funcției formează ieşirile din funcție. Acestea constau din mopdificările efectuate asupra datelor globale, modificări efectuate prin intermediul parametrilor de tipul pointer, valoarea întoarsă de funcție sau alte acțiuni, ca de exemplu afişarea unui mesaj pe monitorul calculatorului.

Fiecare funcție C are un nume precedat de un cuvînt cheie care desemnează tipul funcției. Numele funcției este urmat de o pereche de paranteze rotunde "( )" între care se specifică tipul şi numele parametrilor funcției. Menționăm, că există funcții, care nu au parametri sau care nu furnizează nici un rezultat. O astfel de funcție se specifică prin folosirea cuvîntului cheie **void** luat în paranteze, respectiv în fața numelui ce desemnează funcția.

Orice program scris în limbajul C, indiferent de gradul lui de complexitate, trebuie să conțină o funcție cu numele **main**. Cel mai simplu porgram C este:

```
void main (void)
  {
  }
```

şi constă din funcția **main** al cărei corp, delimitat de perechea de acolade, este gol. Fiind corect din punct de vedere sintactic, programul poate fi compilat, link-editat, lansat în execuție.

Pentru ca programul să capete consistență, adică să execute anumite operații, corpul funcției **main** trebuie completat cu instrucțiuni. De exemplu:

```
#include <stdio.h>
#include <conio.h>
void main (void)
  {
     clrscr( ); printf("Salut prieteni ! Sunt Borland C");     getch( );
  }
```

După cum se poate constata acest program conține 3 instrucțiuni care apelează 3 funcții (**clrscr**, **printf**, **getch**) din biblioteca de funcții a mediului Turbo C++/Borland C.

În limbajul C, pentru a marca sfîrsitul unei instrucțiuni se utilizează caracterul ";". Menționăm că limbajul C ignorează spațiile albe de tipul linie nouă, spațiu şi tabulatorul, cu excepția cazului în care acestea sunt părți ale unui şir de caractere, luate în ghilimele. Aşadar, utilizatorii limbajului C pot să-i organizeze cum doresc textul programului de calcul. Se recomandă a scrie instucțiunile pe linii separate, iar acoladele să fie scrise pe cîte o linie în aceiasi coloană.