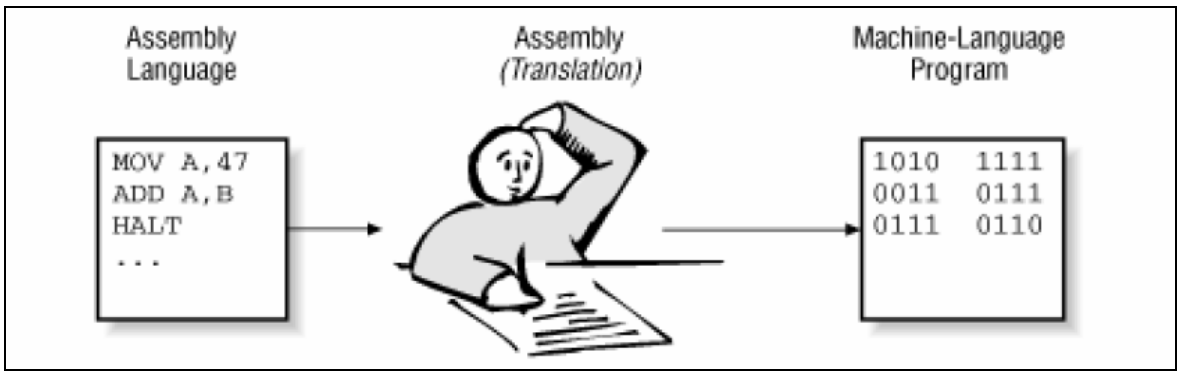
1 ВОВЕД

1. Општ развој на програмските јазици

Со појавата на првите компјутери се појавува потребата од пишување на програми преку кои ќе функционираат деловите од компјутерот. Во тоа време програмерите програмираа на многу неприроден начин, комбинирајќи низи од 0 и 1, користејќи го т.н. **машински јазик**.

Подоцна програмерите согледуваат дека од секој блок на 0 и 1 може да се искомбинираат наредби-зборови кои ќе се многу поразбирливи и полесно ќе се применуваат во текот на пишувањето на програмите. Ваквиот програмски јазик е познат како **assembler**.



Со тек на времето се појавуваат т.н. **програмски јазици на високо-ниво**, кои му нудат на програмерот множество на инструкции кои се лесно разбирливи, а исто така се доволно прецизни и едноставни за компјутерот да може да ги разбере (овде спаѓаат FORTRAN, COBOL, PASCAL…).

Во 1970 година програмерот, Dennis Ritchie, го креира новиот програмски јазик наречен C. (Името го измислил според тоа што овој нов јазик го надградува програмскиот јазик кој претходно го користел – наречен B).

C бил почетно дизајниран за пишување на оперативни системи. Јазикот бил екстремно едноставен и флексибилен, така да подоцна се користи за пишување на најразлични програми. Поради овие причини јазикот станува најпопуларен програмски јазик во светот.

Идејата за измислувањето на програмскиот јазик C е давањето на слобода на програмерот при организацијата и пишувањето на програмата, односно да го напише кодот на начин кој е разбирлив за него, а и за останатите програмери. По

пишувањето на програмата се користи **компајлер** кој ја преведува програмата во машински код кој е лесно разбирлив за компјутерот.

2. Основни елементи на програмскиот јазик C

Секој програмски јазик на високо ниво се состои од множество на резервирани зборови, а комбинацијата од еден или неколку клучни зборови дава наредба од програмскиот јазик.

Множеството на клучни зборови од програмскиот јазик C е следното (32 според ASCII стандардот, 28 според Richie).

auto	**double**	**int**	**struct**
break	**else**	**long**	**switch**
case	**enum**	**register**	**typedef**
char	**extern**	**return**	**union**
const	**float**	**short**	**unsigned**
continue	**for**	**signed**	**void**
default	**goto**	**sizeof**	**volatile**
do	**if**	**static**	**while**

Програмскиот јазик C е функционален јазик, односно кодот напишан во овој програмски јазик е базиран на функции. Основниот облик на програма напишана во програмскиот јазик C е следниот:

Програма во C	**Програма во Pascal**

`void main()`
`{`
`  deklaracija_na_promenlivi;`
 	
`  programski_naredbi;`
`}`
	`Program ime_na_programata;`
	`var deklaracija_na_promenlivi;`
	`begin`
	
	`  programski_naredbi;`
	`end.`

`main` е функција која означува главна програма. Во овој специјален тип на функција може да се декларираат променливи, нови под-функции, и да се пишува кодот на програмата.

Функциите може да примаат влезни параметри. Ова се обезбедува преку **()** делот по името на функцијата. Во овој случај празните загради означуваат дека оваа функција не прима влезни параметри.

Типот на резултатот кој го враќа функцијата стои пред името на функцијата. Во овој случај резервираниот збор **void** означува дека функцијата не треба да врати резултат.

Телото на секоја функција започнува со **{**, а завршува со **}**. Наредбите кои се употребуваат меѓусебно се одвојуваат со **;** - точка и запирка.

За дополнително појаснување на некој програмски сегмент многу често се користат коментарите. Во програмскиот јазик С коментарите се задаваат преку користење на **/*** … ***/** конструктот или преку **//**… конструктот.

```
void main()
{
   deklaracija na promenlivi;
   naredba 1;   /* komentar 1 */
   naredba 2;   /* komentar 2 */
   naredba 3;   // komentar 3
}
```

Пример 1*: Програма Dobredojdovte na ETF!

```
#include <stdio.h>
void main()
{
   printf("Dobredojdovte na ETF!\n");
   /* na ekran ke se ispecati gornata poraka */
}
```

* кратко објаснување на делот include заедно со библиотеката stdio.h
  кратко објаснување на наредбата printf

Од овде се гледа дека основната верзија на структурата на С програмата може да се прошири на:

`INCLUDE секција`	содржи #include изрази за вклучување на надворешни библиотеки, односно да се искористат готови надворешни веќе дефинирани функции
`DEFINE секција`	содржи #define изрази за декларирање на константи и податочни типови
…	дефинирање на глобални променливи и функции
`tip main()`	
`{`	
`tip promenlivi;`	декларација на променливи
`naredbi;`	извршни изрази
`}`	

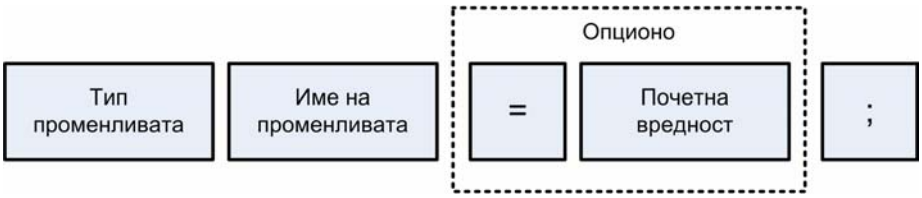
3. Константи

Константите не ја менуваат вредноста при извршувањето на програмата.

- децимални: -1, 0, 12, 189, …
- октални: 015, **019**, 0105, …
- хексадецимални: 0x25, 0xA4C, …
- реални: 3.5F, -2.845F, 1.34e-9, …
- знаковни: ‘a’, ‘b’, ‘c’, ‘\n’, …
- текстуални: “ ”, “Struktuirano programiranje”, …

4. Променливи

Променливите претставуваат места во меморијата каде што ќе се чува некоја вредност. Во C ова место во меморијата се идентификува преку името на променливата. Сите променливи кои ќе се користат во програмата треба претходно да бидат декларирани, а потоа да се користат. Со секое ново сместување на вредност во променливата, старата вредност се брише! Начинот на декларација на променливите е следниот:



Типовите на променливите може да бидат:

Цели броеви	Знаковни	Децимални броеви

`int`
`short`
`long`
`unsigned`
	`char`	`float`
		`double`

При дефинирањето на имињата на променливите треба да се запазат следниве правила, односно имињата на променливите треба да содржат:

- мали букви од **a** до **z**;
- големи букви од **A** до **Z**;
- цифри од **0** до **9**;
- знак за подвлекување _ кој се третира како буква;
- најчесто должината на имињата на променливите е 32 знаци;
- C ги разликува малите и големите букви!

Пример 2: Да се напише програма во C која ќе ја пресмета сумата на два броја.

```
void main()
{
    int a, b, c;
    a = 5;
    b = 10;
    c = a + b; // c=15
}
```

Како што се гледа од примерот освен делот за декларирање на променливите, останатиот дел од програмата ги содржи наредбите кои ја извршуваат бараната задача. Бидејќи се работи за математичка операција, користени се оператори за работа со декларираните променливи.

5. Оператори

Операторите се користат за градење на изрази, при што операциите се изведуваат од лево надесно со што се применува правилото на приоритет на операторите во нивното изведување.

Аритметички оператори:

Се применуваат над бројни променливи (цели броеви или децимални броеви).

+	собирање на два броја
-	одземање на два броја
*****	множење на два броја
/	делење на два броја
%	остаток при делење на два цели броеви

Релациони оператори:

Се применуваат над било кои споредливи типови на податоци, а резултатот е цел број 0 или 1. 0 – неточно, 1 – точно.

<	помало
<=	помало еднакво
>	поголемо
>=	поголемо еднакво
==	еднакво
!=	различно

Логички оператори:

Се користат најчесто во комбинација со релационите оператори за формирање на сложени логички изрази, кои повторно враќаат резултат 0 или 1.

&&	И
||	ИЛИ
!	НЕ

Пример 3. Работа со промеливи и доделување на вредност.

```
void main()
{
   int a;
   float p;
   p=1.0/2.0;      /* p=0.5 */
   a=5/2;          /* a=2 */
   p=(1/2)+(1/8); /* p=0.0 */
   p=3.5/2.8;      /* p=1.25 */
   a=p;            /* a=1 */
   a=a+1;          /* a=2 */
}
```

6. Печатење на податоци (излезни функции)

Како што беше покажано на првиот пример “Dobredojdovte na ETF!”, се користи функција за испраќање на некој текст кон стандардниот уред за испис на податоци – мониторот.

Бидејќи C не вклучува наредби за влез и излез на податоци, се користи библиотека со функции која претходно мора да се пријави во програмата. Тоа се прави со користење на:

#include<stdio.h> *stdio – standard input output*

На овој начин пред да се искомпајлира програмата, предпроцесорот на C знае дека треба да ги вклучи функциите од библиотеката stdio.h со цел да не се јави грешка при компајлирањето. За печатење се користи функцијата printf:

```
int printf(kontrolna_niza,lista_na_promenlivi);
```

Контролната низа содржи било каков текст за испис и контролни знаци предводени од **%** или \. Контролните знаци зависат од видео на променливата чија вредноста треба да се испише или од саканата акција што треба да биде превземена.

Во следната табела се прикажани сите контролни низи:

контролна низа	**објаснување**
%d	за цели броеви
%i	за цели броеви
%c	за знаци
%s	за низа од знаци

%e	реален број во технички формат (e)
%E	реален број во технички формат (E)
%f	реален број во децимален формат
%g	реален број во пократкиот од форматите %e и %f
%G	реален број во пократкиот од форматите %E и %f
%u	цел број без предзнак
%o	октален цел број без предзнак
%x	хексадецимален цел број без предзнак (мали букви)
%X	хексадецимален цел број без предзнак (големи букви)
%p	прикажува покажувач
%n	бројот на испишани знаци се доделува на аргументот
%%	испишување на знакот %

Во низата на променливи покрај променливи може да има и константи и аритметички изрази.

Пример 3. Примена на printf функцијата.

```
#include<stdio.h>
void main()
{
    printf(", brojot na znaci e %d",printf("abcd"));
}
```

```
abcd, brojot na znaci e 4
```

7. Задачи

1. Да се напише програма која ќе ја пресметува вредноста на математичкиот израз:

x = 3/2 + (5 – 46*5/12)

```
#include <stdio.h>
void main()
{
     float x;
     x = 3/2 + (5-46*5/12);
     printf("Vrednosta na x e %f\n", x);
}
```

2. Да се напише програма која за различни вредности на x ќе го пресмета и испечати x^2.

```
#include <stdio.h>
void main()
{
     int x, kvadrat;
     x = 25;
     kvadrat = x*x;
     printf("%d na kvadrat e %d.\n", x, kvadrat);
}
```

3. Да се напише програма која за дадени страни на еден триаголник и ќе ги испечати периметарот и квадратот од плоштината (нека се работи со a=5, b=7.5, c=13.2).

```
#include <stdio.h>
void main()
{
     float a=5.0, b=7.5, c=13.2;
     float L, P, s;
     L = a + b + c;
     s = L/2;
     P = s*(s-a)*(s-b)*(s-c);
     printf("Plostinata na kvadrat e: %f\n", P);
     printf("Perimetarot e: %f\n", L);
}
```

4. Да се напише програма која за дадени страни на еден правоаголник ќе ги испечати неговите плоштина и периметар (пр. a=7, b=10).

```
#include <stdio.h>
void main()
{
     int a, b;
     int L, P;
     scanf("Vnesete gi stranite na pravoagolnikot:", &a,&b);
     L = 2*a + 2*b;
     P = a * b;
     printf("Plostinata e: %f\n", P);
     printf("Perimetarot e: %f\n", L);
}
```

5. Да се напише програма за пресметување на аритметичката средина на броевите 3, 5 и 12.

```
#include <stdio.h>
void main()
{
     int a=3;
     int b=5;
     int c=12;
     float as;
     as = (a + b + c)/3;
     printf("Aritmetickata sredina e: %f\n", as);
}
```

6. Да се напише програма која ќе ги испечати на екран остатоците при делењето на бројот 19 со 2, 3, 5 и 8.

```
#include <stdio.h>
void main()
{
     int a=19;
     printf("Ostatokot pri delenjeto so 2 e: %d\n", a%2);
     printf("Ostatokot pri delenjeto so 3 e: %d\n", a%3);
     printf("Ostatokot pri delenjeto so 5 e: %d\n", a%5);
     printf("Ostatokot pri delenjeto so 8 e: %d\n", a%8);
}
```