5. ФУНКЦИИ

ВЪВЕДЕНИЕ

Функциите са основни структурни единици, от които се изграждат програмите в C++. Всяка функция се състои от множество оператори (възможно е това множество да е празно), които се изпълняват като една обобщена операция. Всяка вече дефинирана (създадена) функция може да бъде извиквана (активирана) и изпълнявана многократно. Всяко извикване на функция е съпроводено с нейното изпълнение. Изпълнението може да стане с различни данни и те се задават при извикването на функцията. В резултат от изпълнението си, функцията може да придобие стойност, която се нарича стойност на функцията или върната стойност.

Всяка C++ програма се състои от една или повече функции. *Сред всички функции на една програма задължително трябва да има една и само една функция с име main.*
Тя определя входната точка в програмата, т.е. тя е първата функция, която се изпълнява при стартиране на програмата.

Добрият стил на програмиране изисква програмите да се изграждат като множество от неголеми функции. Това е свързано със следните предимства:

- програмата става по-прегледна и по-ясна;

- по-лесно става тестването, настройка и модификация на програмата;

- програмата става по-кратка, тъй като многократното повторящите се фрагменти от нея се обособяват като функции, които се дефинират еднократно, и след това се

извикват многократно;

- постига се икономия на памет, тъй като кодът на дадена функция се съхранява само на едно място в паметта, независимо от броя на нейните изпълнения;

- разработването на всякя функция от една голяма програма може да се възложи на отделен програмист и по този начин да се съкрати времето за разработването й;

-

5.1. ДЕФИНИЦИЯ НА ФУНКЦИЯ

Дефиницията на функцията се състои от две части: заглавна част (прототип) и тяло. Общият вид на дефиницията на функция е следния:

[][]

{

тяло (множество оператори)

}

Заглавната част на дефиницията съдържа следните елементи: модификатор, тип на функцията, име на функцията и списък на формалните параметри. Списъкът на формалните параметри се състои от типовете и имената на параметрите, разделени

със запетаи, и е ограден в обикновени скоби.
Тялото на функцията се състои от множество оператори, поместени между две фигурни скоби.
Ето как изглежда дефиницията на функцията sum, чиято върната стойност е сумата на нейните параметри:

```
inline int sum(int x,int y)

{

return x+y;

}
```

Тази дефиниция означава, че sum е име на функция, която връща стойност от тип int и има два параметъра с имена x и y, които са от тип int. Ключовата дума inline е модификатор. Освен него могат да бъдат задавани и други модификатори като overload, virtual, static и пр. Тяхното действие е изяснено по-нататък в тази глава.

Активирането (извикването) на функциите става чрез тяхното име, като формалните параметри се заменят с фактически, например:

```
sum(2,3);

m=3+sum(2,3);
```

В този фрагмент функцията sum е извикана два пъти. При първото извикване няма да има никакъв външен ефект, тъй като върнатата стойност на функцията не се използва.

При второто извикване, функ-цията sum участвува в аритметичен израз. Нейната върната стойност ще бъде добавена към числото 3 и резултатът ще бъде присвоен на променливата m. Следователно m ще получи стойност 8.

Най-често функциите нямат модификатори, т.е. такива не се указват в заглавната част на дефинициите им. Освен това, типът на върнатата стойност на функциите по подразбиране е int. Следователно, ако една функция връща стойност от тип int, в нейната дефиниция може да не се задава тип на върнатата стойност. Имайки предвид тези особености, функцията sum може да се дефинира по следния начин:

```
sum ( int x, int y )

{

return x + y;

}
```

Програма 5.1. Най-проста програма с функция. Функцията намира сумата на двата параметъра.

```
#include

#include

//Дефиниция на функцията sum
```

```
sum(int x,int y)

{

return x+y;

}

//Дефиниция на функцията main

void main()

{

cout
```