Език С. Функции. Деклариране и прототип на функциите. Видове според типа на резултата. Рекурсивни функции. Пример за използването им.

1. Функции:

Основната програмна единица в езика за програмиране *С* се нарича *функция*. Понятието *функция* в езика *С* е аналогично на понятията *подпрограма*, *процедура* и *функция* в другите езици за програмиране он мидоско ниво като *FORTRAN*, *PASCAL*, *ALGOL*. Във функционално отношение тя е част от програма, с изпълнението на която се получават определени резултати. Като програмна единица *функцията* се оформя по строго фиксирани правила и представлява самостоятелен фрагмент от програмата. Този *фрагмент* може да се изпълнява мнгократно в различни части от програмата, като към него се прави обръщение по име.

Чрез използване на *функции* една програма може да се раздели на части (*модули*) и при нейното разработване да се приложат принципите на структурното програмиране. Предимствата на този подход са:

отделните модули могат да се програмират независимо един от друг;

- модулите по-лесно се тестват за грешки и модифицират;

- могат да се използват готови програми, оформени като функции;

- логиката на цялата програма става по-разбираема.

От това, какви свойства имат основните програмни единици и как е организирана работата с тях, до голяма степен се определят възможностите за структурното програмиране. Не без основание, езикът за програмиране *C* и неговите развития *C++* се причисляват към добре структурираните езици, превъзхождащи дори *PASCAL*.

2. Деклариране и прототип на функциите:

Функцията е набор от последователно записани оператори на езика, които осъществяват действия, представляващи логичесдка цялост. Този набор от оператори притежава име, представляващо *идентификатор*. Името на главната функция задължително е *[main]*, а останалите функции получават имена от програмиста или в зависимост от предназначението си, но имена на валидни *идентификатори*.

По-общият синтаксис на *функция* е следният:

тип_на резултата *име_на_функцията (списък_на_параметрите)*

описание на типа на параметрите

{

описание на локалните променливи;

оператори;

.....

return (израз);

.....

оператори;

.....

return (израз);

.....

оператори;

}

Последователно ще разгледаме елементите на това описание, като изясним тяхното предназначение.

Функцията, като вече беше посочено, е самостоятелен фрагмент от програма, съдържащ описания на променливи и набор от оператори на езика. Те се затварят между фигурни скоби и се наричат *тяло на функцията*. В тялото на функцията могат да се използват всеки допустим за езика оператор и описание. Описанията на променливите трябва да се запишат веднага след отварящата скоба в тялото на функцията.

В езика за програмиране *C* е приета технологията на структурното програмиране, известна като *програмиране отгоре надолу*.

За да се изпълнят операторите, които съставят една функция, на нея трябва да ѝ се предаде управлението от друга функция. Процесът на предаване на управлението между функциите се нарича *обръщение към функция* или *извикване на функция*. *Функц ията* ,
която предава управлението, е
извикваща
, а тази, която приема управлението –
извиквана
:

Така може да се дефинира, че в езика за програмиране *C* програмата е съвкупнвост от функции, които предават управлението помежду си.

Програмистът, както вече беше посочено, оформя една функция с име *[main]*, наречена *главна функция*
. При стартирането на изпълнението на програма на езика за програмиране
C

, операционната система (
ОС
) предава управлението на
главната функция
, която от своя страна може да се обръща към други функции. Изпълнението на програмата завършва с изпълнението на
главната функция
, която връща управлението на
ОС
. Като изключение от описания механизъм на взаимодействие между
ОС
и функциите в програми, написани на езика
С
е предвидена възможност за изход от всяка функция направо към
ОС
чрез обръщение към стандартната функция
[
exit
]
фиг.2

Обръщение към функция се извършва чрез името й, което е *идентификатор*, съставен по правилата на езика. Името трябва да не се повтаря в програмата и не може да е ключова дума.

Функцията, като самостоятелна програмна единица има смисъл само ако има възможност да получава и да предава информация от и към другите функции, или към външни устройства. За да се предаде в извикваната функция стойността на една променлива, дефинирана в извикващата функция, е необходимо тази променлива да се включи в списъка на предаваните стойности. Този списък се нарича *списък на аргументите*
или
списък на параметрите на функцията
.

Името на функцията, *списъкът на параметрите* и *описанията на параметрите* оформят *за*

главието
на
функцията
.
Параметрите
, оформени в
списък
, са изброяване на имената на предаваните променливи. Променливите се записват една след друга, опделени със запетая, а целият
списък
се затваря в кръгли скоби и се записва непосредствено след името на
функцията
.

Обикновено, след изпълнението на дадена *функция*, в извикващата *функция* се връщат резултатите от изпълнените действия. Тези резултати могат да се върнат, като се използват:

1. Специалният оператор [return] – тук параметърът трябва да е описан в заглавието на функцията,

2. Глобални п роменливи,

3. Параметрите на функцията.

3. Видове функции според типа на резултата:

Връзките, които програмата реализира между *функциите*, са показани на фиг.3

Виж фиг.3 /Фиг.3 е изнесена в папка „ Фигури към въпрос 10”/

От *главната функция*, според типа на резултата, се различават следните *функции*, показани на горната фигура:

1. Функцията [div] не връща резултат. Стойността на най-големия общ делител се отпечатва във функцията;

2. Функцията [mnum] връща резултат от тип *[int]* – максималното от две числа от тип *[int]*
;

3. Функцията [ekran] няма параметри;

4. Функцията [printf], вече разгледана в предна тема;

5. Функцията [scanf], вече разгледана в предна тема.

Променливите са два вида: *глобални* и *локални* – според областта на действието им.

Един *блок* може да съдържа друг блок и т.н. В такива случаи се казва, че блоковете са вложени един в друг.

4. Рекурсивни функции. Пример за използването им:

Едине обект е *рекурсивен*, ако се съдържа в себе си или е определен с помощта на себе си. Принципът на рекурсията е мощен подход за изграждане на алгоритми, намиращ широко приложение в редица области. Езикът за програмиране
C
поддържа две езикови конструкции, които позволяват да се реализират:

Рекурсивни функции, и

Структури с рекурсии.

1. Рекурсивни функции:

В езика за програмиране *C* е допустимо описанието на функция да се съдържа обръщение към самата себе си:

```
int fun(a,b)

 int a, b;

 {

 int i;

 ……

 i = fun(a-1, b-1);
```

```
……

}

main()

{

int x, y, c;

……

……

c = fun(x, y)

……

}
```

Такава *функция* се нарича рекурсивна. Използването ѝ не изисква от програмиста допълнителни описания или действия. В компилатора на езика за програмиране *C* *рекурсията* се изпълнява / представлява от последователни, но крайни по брой обръщения към функция. Затова при всяко обръщение към

функцията
, компилаторът
извършва обичайните действия: записва в стека адреса за връщане, параметрите на функцията и локалните променливи. Този пример се изпълнява като запис в стека показан на : фиг.4

BP – Base Point – Указател на базата;

SP – Stack Point – Указател на стека – сочи винаги върха на стека.

От фигурата се вижда, че при всяко рекурсивно обръщение към стека се създава отделно множество за параметрите и локалните променливи и по този начин се разрешава конфликтът между еднаквите *идентификатори*. Правилото е просто: използваните в момента *иден тификатори*
са последните, записани в стека. Разглежданата схема насочва вниманието и към нещо много важно –
рекурсивните
обръщения трябва да бъдат краен брой. Затова задължение на програмиста е да предвиди във функцията управляваща логика, която да спре
рекурсията
.

Пример за това може да бъде математическата функция: