ШАГ компьютерная академия

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ С

Урок №4

Програмиране на език C

Съдържание

1. Конструкция for

В предишния урок ние сме се запознали с такова понятие като цикъл и сме разгледали няколко от конструкциите, които представят цикъла в език С. А именно – while и do while. Сега ще разгледаме още една разновидност на цикъла – оператор for. Дадения оператор теоретически представя пълна аналогия на while а практически позволява да се организира цикъла с по-удобно управление. Общ синтаксис и принцип на работата на конструкцията for

```
for(инициализация на променливата;проверка на условието;промяна на променливата)
{
                действие;
}
```

Принцип на изпълнение на цикъла:

```
1. Инициализация на променливата.
2. Проверка на условието .
3. Изпълняване на действието ако условието е вярно.
4. Ако условието е грешно, изпълнение на следващото след цикъла.
5. Ако условието е било вярно, промяна на управляващата променлива.
6. Проверка на условието. След това отново пункт 3 или 4.
```


вход
СОЗДАНИЕ И ИНИЦИАЛИЗАЦИЯ УПРАВЛЯЮЩЕЙ ПЕРЕМЕННОЙ
проверка условия
true
выполнение тела цикла
false
изменение управляющей переменной
выход

Пример за употреба

Да разгледаме простичък, вече познат пример: с помощта на цикъла да покажем на екрана цифрите от 1 до 5 включително. Само че ще го направим с помощта на оператора for.

```
#include <iostream>
void main()
{
            for(int i=1;i<=5;i++)
            {
                cout<<i;
            }
}
```

Коментар към примера.

1. Вътре в цикъла се обявява променлива i равна на 1. Тя ще бъде управляващата променлива.

2. След това се провежда проверка на значението на тази променлива с помощта на условието i<=5;

3. Ако условието е вярно (а това ще е така докато i не достигне значение 6) се показва значението на i на екрана (count<<i;) и промяна на управляващата променлива i с 1 (i++). След това отново се проверява условието.

2. Ако условието е грешно (тоест значението на i е станало равно на 6) програмата преминава към следващия ред зад закриващата фигурна скоба цикъла.

Забележка: Обърнете внимание, че първата стъпка — СЪЗДАВАНЕ И ИНИЦИАЛИЗИРАНЕ НА ПРОМЕНЛИВА — винаги се изпълнява само един път.

Някои особености на синтаксиса for

Независимо от елементарността на работата на оператора, той притежава няколко особености на форми за записване.

Инициализация на управляващата променлива

1. Инициализация и създаване на променлива проведено в цикъла.

```
for(int x=1;x<=100;x++)
{
          cout<<x;
}
```

2. Създаване на променлива проведено преди цикъла, а инициализация в цикъла.

```
int x;
for(x=1;x<=100;x++)
{
               cout<<x;
}
```

3. Инициализация и създаване на променлива проведена в цикъла.

```
int x=1;
for(;x<=100;x++)
{
        cout<<x;
}
```

Всичките три примера са абсолютно функциониращи и равносилни.

Промяна на управляващата променлива.

Промяната на управляващата променлива може да се пренесе в тялото на цикъла, как се случва това в while и do while.

```
for(int x=1;x<=100;)
{
        cout<<x;
        x++;
}
```

Условие.

Условието на конструкцията може да бъде пропуснато, обаче в такъв случай то ще се подразбира автоматично за вярно. По такъв начин ние получаваме винаги вярно условие и както следва – ВЕЧЕН ЦИКЪЛ.

```
for(int x=1;;x++)
{
            cout<<x;
}
```

Забележка: Ако искате да разберете как да пропуснете условието и да избегнете вечен цикъл – прочетете следващия раздел на урока.

Изхождайки от гореописаното, можем да направим следния извод: **Нито една от частите на цикъла for не е задължителна.**

Както виждате, работата на for е проста и аналогична на работата на while?! Това зависи от поставената задача и от вашето решение.

Оператор break

Доста често при работата ни с цикъла, възниква необходимост изкуствено да се спре изпълнението на цикъла. За това се използва вече познатият ви (от изучаването на switch), оператор break. Този оператор трябва да се намира в тялото на цикъла, в това място където трябва да се спре. Например, именно с помощта на този оператор, ние можем да решим проблема с вечния

цикъл, в ситуации когато условието на for не е указано в цикъла.

```
#include <iostream>
using namespace std;
void main()
{
              for(int x=1;;x++)
              {
                  if(x==4) break;// ако x е станал равен на 4 – спиране на
                  цикъла cout<<x;

              }
              cout<<"Bye!";
}
```

Коментар към примера

1. Съгласно правилото, условието на цикъла винаги е вярно, защото него просто го няма

2. При значенията 1,2 и 3 променливата x условие на оператора if няма да се изпълнява. break, естествено няма да сработи, след като се намира в тялото на if. Междувременно на екрана последователно ще излизат числата 1,2,3.

3. Когато x стане равно на 4, програмата попада в тялото на if и се изпълнява break. Цикъла веднага ще бъде спрян, а изпълнението на програмата ще премине към следващия етап зад закриващата оператора for фигурна скоба.

4. На екрана се появява надпис Bye!

5. Цифрата 4 на екрана никога няма да излезе след като ако е сработил оператора break, всичко което се намира под него не се изпълнява.

Забележка: break може да бъде използван в цикъла или в оператора switch. На всяко друго място би довело до грешка на етапа за компилиране.

Оператор continue

Оператор continue се използва за прекъсване на текущия процес от цикъла и преминаване към следващата стъпка. В много случаи такива действия са необходими. Ако се изпълни оператор continue, то в зависимост от вида на цикъла произлиза следното:

Цикъл while и do while спират изпълнението на стъпката и преминават към проверка на условието.

Цикъл for също спира изпълнението на стъпката. Но първо преминава към промяна на управляващата променлива и след това към проверка на условието.

Да разгледаме пример: да се изкарат на екрана всички нечетни числа в диапазона от нула до 25 включително.

Наименование на проекта Odd.

```
#include <iostream>
using namespace std;
void main()
{
	for(int i=0;i<26;i++)

	{
	 if(i%2==0)// ако числото се дели на две без остатък
	 {
			continue;// да се спре итерацията на цикъла да се мине на i++
	 }
	 cout<<i<<"\n";

	}
}
```

Коментар към примера

1. Цикъла започва своето движение от нула и преминава до 25 включително.

2. В цикъла е предвидено условието: ако числото i - е четно, трябва да се спре текущата стъпка на цикъла (continue;) и да се премине към конструкцията i++.

3. Това, което се разполага под сработилия оператор continue на текущата стъпка вече няма да се изпълни.

4. Ако условието if не се изпълни, значи числото i е нечетно, if ще бъде игнориран, а числото изкарано на екрана.

Сега, когато ние сме се запознали с теоретичните материали на урока, нека да преминем към следващия раздел, където ще бъдат разгледани няколко практически задачи.

2. Практически примери

Пример 1

Постановка на задачата

Часовника звъни всеки час, толкова пъти, колкото е времето. Напишете програма, която да пресметне, колко пъти, ще прозвъни часовника за 12 часа. Име на проекта Time.

Код за реализация

```
#include <iostream>
using namespace std;
void main(){
      int sum=0;
      for(int bom=1; bom<=12;bom++){
        sum+=bom;// насъбиране на сумата звънене
      }

      // Часы пробили 78 раз.
      cout<<" Hours have punched "<<sum<<"times.\n\n";
}
```

Коментар към кода.

1. Изначално се обявява променлива sum която е равна на нула.

2. Цикъла се формира от три конструкции int bom=1; — начална инициализация, bom<=12; — условие, bom++ — промяна на управляващата променлива.

3. В тялото на цикъла се насъбира сумата звънене чрез прибавянето на управляващата променлива към значението на общата сума.

4. Когато i достигне значение 13, цикъла ще спре и на екрана ще се покаже резултата.

Пример 2

Условие на задачата:

Потребител с клавиатура последователно въвежда цели числа. Когато потребителя въведе 0, е необходимо да се покаже на екрана сумата на всички въведени числа. Име на проекта Amount.

Код за реализация:

```
#include <iostream>
using namespace std;
void main(){
	int digit, sum=0;

	for(;;){ // реализация на безкраен пример

	  cout<<"Enter digit:"; cin>>digit;
	  // ввод числа if(digit==0)		//
	  если введен 0
			break;		//спиране на цикъл
	  sum+=digit;		// събиране на сума
	}

	// показване на резултата
	cout<<"Sum of digits"<<sum<<"\n\n";
}
```

Коментар към кода:

1. В програмата е реализиран условно безкраен цикъл. Тоест спирането на цикъла произлиза чрез изкуствен начин (break).

2. На всяка стъпка потребителя въвежда число.

3. Осъществява се проверка, ако това число е 0, значи е време да се спре цикъла, ако не е 0, е необходимо да се прибави към общата сума.

4. След като оператора break отработи и цикъла спре, на екрана ще се покаже сумата на всички въведени с клавиатурата числа.

Пример 3.

Постановка на задачата.

Да се напише програма, която да показва всички числа, на които е кратно числото въведено с клавиатурата. Име на проекта Number.

Код за реализация.

```
#include <iostream>
using namespace std;
void main(){
        int digit;
        cout<<"Enter digit:";
        cin>>digit;

        // цикъла избира числа от 2 до въведеното число
        for(int i=2;i<digit;i++){

        // ако числото не се дели на текущото
        // значение i без остатък да се спре
        // дадената стъпка и да се премине към
        // следващата
        if(digit%i!=0)
                continue;

        // да се покаже i на
        екрана
         cout<<i<<"\n";
        }
}
```

Коментар към кода:

1. Потребителя въвежда число за анализ.

2. Цикъла последователно подбира число от 2 до въведеното.

3. Осъществява се проверка: ако търсеното число не се дели без остатък на текущото е необходимо да се

спре дадената стъпка на цикъла и да се премине към частта i++. (continue).

4. Ако търсеното число се дели на текущото без остатък на екрана ще се покаже текущото число.

3. Домашна работа

Във втория урок сте се научили да разбивате число на цифри. Днешното домашно е основано именно на този принцип, обаче на вас ще ви се наложи да използвате и цикъл.

1. Потребителя въвежда с клавиатурата число – програмата трябва да покаже колко има в даденото число цифри. Числото се записва цяло в една променлива.

Забележка: Например, потребителя въвежда число 12345. На екрана трябва да се появи съобщение за това, че в числото има 5 цифри.

2. Потребителя въвежда с клавиатурата число, да се преобърне това число и да се изкара на екрана.

Забележка: Например, потребителя е въвел числото 12345. На екрана трябва да се появи числото - 54321.

3. Потребителя въвежда с клавиатурата число, да се покаже на екрана сумата от неговите цифри.

Забележка: Например, потребителя е въвел числото 12345. На екранна трябва да се появи съобщение, че сумата на цифрите е 15.