**КОНСТРУКЦИИ ЗА ЦИКЪЛ.**

**Замяна на операторите за инкрементиране и декрементиране**

**Да разгледаме следната задача:**

**Пример 1:**

**Да се състави програма, която въвежда цяло число x и извежда на екарана първите 3 числа, по-големи от x**

**#include <iostream>**

**using namespace std;**

**int main() {**

**int x;**

**cout<<"x=";**

**cin>>x;**

**cout<<x+1<<endl;**

**cout<<x+2<<endl;**

**cout<<x+3<<endl;**

**return 0;**

**}**

**Изход:**

x=9

10

11

12

**Представете си как ще се реши задачата, ако нейното условие изискваше изход не на 3, а на 100 числа? А на 1000?**

**Не е логично да използваме горният начин, тъй като това означава да изпишем най-малко 100 реда код или 1000 реда код. Разгелждайки подробно решението, забелязваме, че има зависимост между операторите, извеждащи числата. Тази зависимост може да се представи като ЦИКЛИЧЕН АЛГОРИТЪМ.**

**Ето как ще изглежда горната задача:**

**#include <iostream>**

**using namespace std;**

**int main() {**

**int x;**

**cout<<"x=";**

**cin>>x;**

**for(int i=1;i<=3;i++) // i=i+1**

**cout<<x+i<<" ";**

**return 0;**

**}**

**Цикълът съдържа 4 основни части:**

* **Инициализация – задава се начална стойност на някои променливи, участващи в цикъла**
* **Тяло на цикъла – съдържа действията, които се повтарят многократно**
* **Актуализация – обновяване стойностите на участващите величини и подготовка за следващото изпълнение на цикъла**
* **Условие за край на цикъла – гарантира прекратяване изпълнението на цикъла**

**Едно изпълнение на тялото на цикъла се нарича итерация**

**За създаване на циклични алгоритми се използват 3 оператора:**

* **FOR – оператор за броячен цикъл**
* **WHILE – цикъл с предусловие**
* **DO…WHILE – цикъл със следусловие**

**1. ЦИКЪЛ FOR**

**Цикъла FOR дава възможност една или повече конструкции в С да се изпълняват многократно. Много програмисти смятат тази конструкция за цикли за най-гъвкава.**

**Тук ще се запознаем с конструкцията FOR в най-общата и форма.**

**for(инициализация; провека на условие; инкрементиране) конструкция**

**инициализация –** задаване начална стойност на променливата използвана за управление на цикъла /променлива за управление на цикъла/, тази секция за инициализация се изпълнява само веднъж преди започване на цикъла

**проверка на условие-** сравнява променливата за управление на цикъла със зададената стойност и ако проверката покаже резултат **true** цикълът се повтаря, ако резултата е **false,** изпълнението на програмата се прехвърля на реда след цикъла

Проверката на условието се извършва в началото или преди всяко повторение на цикъла

**инкрементиране**- тази част се изпълнява след цикъла, тоест след като блокът конструкции формиращи тялото на цикъла са се изпълнили, **предназначението на инкрементацията е да увеличи или намали променливата за управление на цикъла с определена стойност !!!**

**Пример 2.**

**Ще решим Пример 1 чрез цикъл for, но не конкретно за първите 3 числа, по-големи от x, а за произволен брой такива. Колко да бъдат те, ще се зададе чрез променливата n в програмата.**

**#include <iostream>**

**using namespace std;**

**int main() {**

**int x,n;**

**cout<<"x=";**

**cin>>x;**

**cout<<"n=";**

**cin>>n;**

**for(int i=1;i<=n;i++)**

**cout<<x+i<<endl;**

**return 0;**

**}**

**Изход:**

x=17

n=6

18

19

20

21

22

23

**Цикълът for основно се използва за индуктивни циклични процеси, а променливата i от горният пример се нарича управляваща променлива на цикъла. Тя служи предимно за задаване броя на повторенията на цикъла. Тя служи предимно за задаване броя на повторенията на цикъла. В много от задачите обаче, тази променлива участва и в тялото му. Ето един пример за това е следната задача:**

**Пример 3: Да се състави програма, която въвежда от клавиатурата естествено число n и извежда всички естествени числа, по-малки от n и кратни на 5.**

**#include <iostream>**

**using namespace std;**

**int main() {**

**unsigned int n;**

**cout<<"n=";**

**cin>>n;**

**for(int i=1;i<n;i++)**

**if(i%5==0)cout<<i<<endl;**

**return 0;**

**}**

**Изход:**

n=15

5

10

**Пример 4: Да се състави програма, която въвежда от клавиатурата петцифрено естествено число k и го извежда в обратен ред на цифрите му.**

1. **Последната цифра на k се отделя като остатък при деление на k с 10 и се извежда на екрана.**
2. **Посредством целочислено деление на k с 10 се получава нова стойност на k, която има една цифра по-малко**
3. **Предните две действия се повтарят 5 пъти, колкото са цифрите на числото**

**#include <iostream>**

**using namespace std;**

**int main() {**

**unsigned int k;**

**cout<<"k=";**

**cin>>k;**

**for(int i=1;i<=5;i++)**

**{**

**cout<<k%10; //извежда последната цифра**

**k=k/10; //премахва последната /изведена вече/ цифра**

**}**

**return 0;**

**}**

**Изход:**

k=42196

69124

**Пример 5: Програма, която използва цикъла for за да изпише числата от 1 до 10/ през 1, 2, 3 и т.н/**

**int main(void)**

**{**

**int i;**

**for(i=1; i<11; i=i+2) //инициализац., пров.условие.,** инкрементация

**cout<< i<<” “;**

**return 0;**

**}**

**Резултат:**

1 2 3 4 5 6 7 8 9 10 krai

Програмата работи по следния начин:

1. променливата num се инициализира така че да бъде равна на 1 /това става само веднъж и то сега/ num=1
2. проверява се израза num<11

ако num<11 , т.е. true се стартира цикъла for, изписва се числото 1 и num се инкрементира /в случая числото се увеличава с 1/ **num=num+1.**

Това продължава докато num=11.

1. Когато num=11 цикъла for спира и на екрана се изписва terminating

Ако със започване на проверката резултата е false цикъла няма да се изпълни нито веднъж

**Пример 6:**

**int main(void)**

**{**

**int num;**

**for(num=11; num<11; num=num+1) cout<< num; //цикъла няма да се изпълни**

**cout<<"terminating";**

**return 0;**

**}**

Резултат:

terminating

Това е така защото променливата 11 е инициализирана със стойност 11 което прави условието за проверка грешно

**Пример 7:**

**int main()**

**{**

**int i;**

**for (i = 0; i < 3; i++)**

**{**

**cout<<"Hello"<<endl;**

**cout<<"World";**

**}**

**return 0;**

**}**

Резултат:

Hello

World

Hello

World

Hello

World

**Пример 8:** Тази програма повтаря няколко конструкции като използва **блок с код**

Тази програма изчислява сумата и произведението на числата от 1 до5

**int main(void)**

**{**

**int num, sum, prod;**

**sum = 0;**

**prod = 1;**

**for(num=1; num<6; num=num+1) { //инкрементация с 1-ца**

**sum = sum + num; //увеличение на сумата – текуща плюс инкрементацията**

**prod = prod \* num; //увеличение на текущо произведение с инкремент.число**

**}**

**cout<<"product and sum:"<<prod<< sum;**

**return 0;**

**}**

**Резултат:**

product and sum: 120 15

Проверка: 1+2+3+4+5 = 15 //това е сумата

1\*2\*3\*4\*5 = 120 //това е произведението

Цикълът for може да се изпълнява и низходящо. Например :

**for(num=20;num>0;num=num-1)**

Тук виждаме декрементиране /намаление/ на променливата

Освен това декрементацията или инкрементацията може да става и с повече от 1-ца.

**Пример 9: Тази програма брои до 100 през 5**

**int main(void)**

**{**

**int i;**

**for(i=0; i<101; i=i+5) cout<<i;**

**return 0;**

**}**

**Резултат:**

0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100

**Пример 10 : Програмата преобразува десетични числа в двоични**

**int main()**

**{**

**int n, c, k;**

**cout<<"Enter an integer in decimal number system";**

**cin>>n;**

**cout<<"in binary number system is: "<<n;**

**for (c = 31; c >= 0; c--)**

**{**

**k = n >> c;**

**if (k & 1)**

**cout<<"1";**

**else**

**cout<<"0";**

**}**

**cout<<" ";**

**return 0;**

**}**

**Резултат:**

**Enter an integer in decimal number system**

**512**

**512 in binary number system is:**

**00000000000000000000001000000000**

**Пример 11: Тази програма определя дали дадено число е просто**

**int main(void)**

**{**

**int num, i, is\_prime;**

**cout<<"Enter the number to test: ";**

**cin>>num;**

**is\_prime = 1;**

**for(i=2; i<=num/2; i=i+1) /\*ако 2<= (въведеното от нас число разделено на 2)нашето число не е просто\*/**

**if((num%i)==0) is\_prime = 0;**

**if(is\_prime==1) cout<<"The number is prime.";**

**else cout<<"The number is not prime.";**

**return 0;**

**}**

Резултат:

Enter the number to test: 1

The number is prime.

Enter the number to test: 4

The number is not prime.

......................................

**2. Замяна на операторите за инкрементиране и декрементиране в С**

В предишната точка за конструкцията for стана въпрос за инкрементация и декрементация примерно: (num=num+1)

Въпреки , че не е грешно, в професионално написаните програми няма да срещнете конструкции като num=num+1

Това е така, защото С осигурява специален оператор, който увеличава променливата с 1-ца. Този оператор е **++** /два плюса/

Така i=i+1 може да бъде променено на **i++**

**for(num=0; num < some\_value; num++)**

**num++ инкрементация**

**num-- декрементация**

Освен, че спестява писане, този начин на инкрементация и декрементация спестява и време, тъй като компилаторът на С избягва отделни машинни инструкции за зареждане и съхранение и в изпълнимата версия на програмата ги заменя с единична инструкция за инкрементиране и декрементиране**.**

Операторите за инкрементация ++ и декрементация – не е задължително да бъдат след променливата. Въпреки, че ефектът върху променливата е един и същ, позицията на оператора оказва влияние върху извършването на операцията.

Ето и примерите които показват тази разлика:

**Пример 12:**

**int main(void)**

**{**

**int i, j;**

**i = 10;**

**j = i++;**

**cout<<"i and j: " << i<<j; /\* ще изкара 11 10 \*/**

**return 0;**

**}**

Резултат:

i and j: 11 10

Как работи програмата:

1. На j се присвоява текущата стойност на 1 /в случая 10/
2. i се инкрементира /i=i+1, i=11/

Ето защо стойността на j e10 , а не 11.

**Когато операторът за инкрементиране или декрементиране е поставен след променливата, тогава съответната операция се извършва след като стойността на променливата вече е извлечена**

**a=10 \* b++**

**/ това присвоява на num стойност 10 и увеличава i с 1-ца/**

**Пример 14: В този случай оператора за инкрементация ++ е преди променливата**

**int main(void)**

**{**

**int i, j;**

**i = 10;**

**j = ++i; /\* this will print 11 11 \*/**

**cout<<"i and j:"<< i<<j;**

**return 0;**

**}**

**Резултат:**

i and j: 11 11

Когато променливата е предшествана от оператор за инкрементация или декрементация, тогава първо се изпълнява съответната операция и след това се извлича стойността на променливата за използване в израза

**Пример 16 :**

**int main() {**

**int x = 2;**

**int z;**

**z=x++;**

**cout<<z<<x;**

**return 0;**

**}**

Резултат:

z=2 x=3

1. **Вариации на цикъла FOR**

В С цикъла FOR е със значително по-големи възможности, отколкото в повечето други компютърни езици.

Причината за по-голяма гъвкавост на FOR е в това, че изразите, наречени инициализация, проверка на условие и инкрементиране не се ограничават до тези тесни роли.

Друга важна причина е, че едни или повече от изразите, които присъстват в него, могат да бъдат празни. Например ако променливата за управление на цикъла вече е инициализирана извън FOR, няма нужда от израз за инициализация.

**Пример 17 :** Тази програма продължава изпълнението на цикъла, докато от клавиатурата въведете **q.** Вместо да проверява променливата за управление на цикъла, проверката на условието в този цикъл for проверява стойността на знака, въведен от потребителя.

**#include<iostream>**

**using namespace std;**

**int main(void)**

**{**

**int i;**

**char ch;**

**ch = 'a'; /\* задава на ch начална стойност \*/**

**for(i=0; ch != 'q'; i++) {**

**cout<<"pass:"<< i;**

**cin>>ch;}**

**return 0;**

**}**

**Резултат:**

pass: 0

Тук условието което контролира цикъла, нищо общо с променливата за управление на цикъла. Причината на **ch** да е зададена начална стойност е, за да се предотврати случайното съдържание на q при започването на програмата.

**Пример 18: Тук променливата за управление на цикъла се инициализира от потребителя извън цикъла, което означава, че частта за инициализация е празна.**

**int main(void)**

**{**

**int i;**

**cout<<"Enter an integer: ";**

**cin>>i;**

**for(; i; i--) cout<< i;**

**return 0;**

**}**

Резултат: Enter an integer: 8

RUN SUCCESSFUL (total time: 1s)

8 7 6 5 4 3 2 1

**Пример 19:** Това е вариант на FOR при който неговата цел може да бъде празна

Например тази програма продължава да чете знакове от клавиатурата, докато потребителя въведе q

**#include<iostream>**

**using namespace std;**

**int main(void)**

**{**

**char ch;**

**for(ch=0; ch!='q'; cin>>ch)**

**;**

**cout<<"Found the q.";**

**return 0;**

**}**

Резултат:

f

t

u

n

q

Found the q.

Използването на FOR може да създаде циикъл, който никога не спира

Този вид цикли обикновено се наричат безкрайни.

Въпреки, че създаването на безкраен цикъл е логическа грешка /бъг/ понякога ще искате да създадете такава умишлено.

За да създадете безкраен цикъл , може да се използва FOR по този начин:

For(; ;){

.

.

.

}

**Пример 20:** В С/C++, за разлика от повечето други компютърни езици, променливата за управление на цикъла може да се променя извън частта за инкрементиране.

Тази програма” ръчно” увеличава i в края на всяка интерация на цикъла.

**int main(void)**

**{**

**int i;**

**for(i=0; i<10; ) {**

**cout<<i;**

**i++;**

**}**

**return 0;**

**}**

Резултат:

0 1 2 3 4 5 6 7 8 9

RUN SUCCESSFUL (total time: 47ms)

**СЪЗДАВАНЕ НА ВЛОЖЕНИ ЦИКЛИ**

Когато тялото на един цикъл съдържа друг, втория цикъл се нарича вложен в първия. Всеки един от циклите в С може да бъде вложен в който и да било друг. Стандартът на ANSI за С определя, че циклите могат да се влагат на дълбочина най-малко 15 нива. Въпреки това повечето компилатори позволяват влагане до практически всякакво ниво.

Като прост пример за вложени цикли for, този фрагмент отпечатва на екрана десет пъти числата от 1 до 10

**for(i=0; i<10; i++) {**

**for(j=1; j<11; j++) cout<< j<<endl; /\* грешен цикъл \*/**

**}**

**Пример 21:** Програмата използва три цикъла for, за да отпечата азбуката три пъти, като всяка буква се изобразява два пъти

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int i, j, k;**

**for(i=0; i<3; i++)**

**for(j=0; j<26; j++)**

**for(k=0; k<2; k++) cout<<'A'+j;**

**return 0;**

**}**

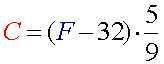
**Резултат:**

AABBCCDDEEFFGGHHIIJJKKLLMMNNOOPPQQRRSSTTUUVVWWXXYY

Конструкцията **cout<< 'A'+j;** работи, защото ASCII кодовете на буквите от азбуката са строго нарастващи –всеки следващ е по-голям от този на предходните букви

**Пример. 22**

**Пример 3: / използва оператора for/** Температурата по Фаренхайт между 0 и 300 градуса, увеличена с 20 (0,20,40,60............300) изкарайте на екрана температурите конвертирани в Целзии



**int main()**

**{**

**int Fahrenheit;**

**for (Fahrenheit = 0; Fahrenheit <= 300; Fahrenheit = Fahrenheit + 20)**

**cout<< Fahrenheit<<"F="<<(5.0/9.0)\*(Fahrenheit-32)<< "C";**

**return 0;**

**}**

**Резултат:**

**0 -17.778**

**20 -6.667**

**40 04.444**

**..................**

**280 137.778**

**300 148.889**

**ДОПЪЛНИТЕЛНИ ЗАДАЧИ**

**Зад.1 Да се състави програма, която въвежда от клавиатурата естествено число n. Програмата да изчислява и извежда:**

**а/ сумата на всички цели числа от 1 до n**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**unsigned long n,sum=0;**

**cin>>n;**

**for(int i=1;i<=n;i++) sum+=i;**

**cout<<sum<<endl;**

**return 0;**

**}**

**б/ произведението на всички цели числа от 1 до n (n!=1\*2\*3\*..\*n се нарича факториел)**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**unsigned long n,fact=1;**

**cin>>n;**

**for(int i=1;i<=n;i++) fact\*=i;**

**cout<<fact<<endl;**

**return 0;**

**}**

**Зад.2 Да се състави програма, която въвежда от клавиатурата естествено число k. Програмата да извежда всички трицифрени числа, сумата от цифрите на които е равна на числото k.**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**unsigned short k,sum;**

**cin>>k;**

**for(int i=100;i<1000;i++)**

**{**

**sum=i%10;**

**sum+=i/10%10;**

**sum+=i/100;**

**if(sum==k)cout<<i<<' ';**

**}**

**return 0;**

**}**

**Зад.3 Да се състави програма, която извежда от клавиатурата целите числа m и n (m<n). Програмата да извежда всички числа в интервала [m,n], които са кратни на 5.**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int m,n;**

**cin>>m>>n;**

**for(int i=m;i<=n;i++)**

**if(i%5==0)cout<<i<<' ';**

**return 0;**

**}**

**Зад. 4 Да се състави програма, която намира и извежда всички трицифрени числа, които нямат в записа си цифра 0 и са кратни на всяка своя цифра.**

**Зад. 5 Да се състави програма, която извежда на екрана всички четирицифрени числа, сумата на цифрите на които е двуцифрено число**

**Зад. 6 Да се състави програма, която намира и извежда всички трицифрени числа, които имат поне две равни цифри**

**Зад. 7 да се състави програма, която въвежда от клавиатурата естествено число n и реално число a. Програмата да извежда аn**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**unsigned int a,n,step=1;**

**cin>>n>>a;**

**for(int i=1;i<=n;i++) step\*=a;**

**cout<<step<<endl;**

**return 0;**

**}**