**Цикъл WHILE, DO, ВЛОЖЕНИ ЦИКЛИ, ЦИКЛИ ЗА ПРЕХОД (BREAK, CONTINUE, SWITCH, GOTO)**

1. **Цикъл WHILE /цикъл с предусловие/**

**while(израз)конструкция**

**Цикълът while работи чрез повтаряне на своята цел, докато изразът е верен.**

**Когато той стане грешен, цикълът спира.** Стойността на израза се проверява в началото на всяка интервенция на цикъла. Това означава, че ако изразът, с който се започва е грешен, цикълът няма да бъде изпълнен нито веднъж.

**Пример 1:**

Ако трябва да решим задачата от цикъл for /за извеждане в обратен ред на петцифрено число/ :

**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;**

**}**

за число с произволен брой цифри, процесът вече е итерактивен, т.е. не се знае броят на повторенията.

Поради тази причина по-удобен за ползване е операторът за цикъл с предусловие, а това става по следният алгоритъм:

1. Последната цифра на k се отделя
2. Посредством целочислено деление на к с 10 се получава нова стойност за k, която има една цифра по-малко
3. Предните две действия се повтарят докато k получи стойност 0. Това е условието за края на цикъла

**#include <iostream>**

**using namespace std;**

**int main() {**

**int k;**

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

**cin>>k;**

**while(k!=0)**

**{**

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

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

**}**

**return 0;**

**}**

Изход;

k=74960382

28306947

**Пример 2:** Чрез използване на while може по-добре да се изчака въвеждането на q.

**#include<iostream>**

**#include <cstdio>**

**using namespace std;**

**int main(void)**

**{**

**char ch;**

**while(ch!='q') cin>>ch;**

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

**return 0;**

**}**

**Резултат:**

q

Found the q.

**Пример 3:** Машина за кодиране. Тази програма трансформира въведените от вас знакове в кодирана форма чрез добавяне на единица към всяка буква. Например ‘A’ става ‘B’ и т.н.

Програмата спира да работи, когато натиснете ENTER

**#include <iostream>**

**#include<cstdio>**

**using namespace std;**

**int main()**

**{**

**char ch;**

**cout<<"Enter your message";**

**cin>>ch;**

**while(ch != '\r') {**

**cout<<(char)(ch+1);**

**cin>>ch;**

**}**

**return 0;**

**}**

**Резултат:**

Enter your message.

w

x  
r

s

**Пример 4:** Сумата на първите 100 числа

**#include<iostream>**

**using namespace std;**

**int main() {**

**int suma = 0 ;**

**int i = 0;**

**while ( i <= 100)**

**{**

**suma += i; //suma=suma+i**

**i++; /\* !!!! без този ред=> безкраен цикъл \*/**

**}**

**cout<<"suma="<<suma ;**

**return 0;**

**}**

**Резултат:**

suma=5050

**Пример 5:** Oбръщане на номера

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int n, reverse = 0;**

**cout<<"Enter a number to reverse "<<endl;**

**cin>>n;**

**while (n != 0)**

**{**

**reverse = reverse \* 10;**

**reverse = reverse + n%10;**

**n = n/10;**

**}**

**cout<<"Reverse of entered number is = "<<reverse;**

**return 0;**

**}**

**Резултат:**

Enter a number to reverse

36

Reverse of entered number is = 63

**Пример 6. Програмата изписва числата от 100 до 0:**

**#include <iostream>**

**using namespace std;**

**int main() {**

**int i;**

**i = 100;**

**while(i >-1) {**

**cout<<i;**

**i--;**

**}**

**return 0;**

**}**

1. **Цикъла DO- WHILE /цикъл със следусловие/**

Последният цикъл в С е Do. Той има следната форма:

**do{**

**конструкции**

**}while(израз);**

Ако се повтаря само една конструкция , фигурните скоби не са необходими. Повечето програмисти все пак ги включват, за да могат по-лесно да разпознават, че думата while, с която завършва цикълът do, е част от цикъла do а не начало на цикъла while

**Цикълът Do повтаря конструкцията /конструкциите/ докато изразът е верен. Той спира да се изпълнява, когато изразът стане грешен. Цикълът DO е уникален, защото изпълнява кода от своето тяло поне веднъж.** Това е така, защото изразът , контролиращ цикъла, се проверява в края на всяка интерация на цикъла

**Пример 7. Да се напише програма, която въвежда от клавиатурата естествено число n и цифра p.**

**Aко в записа на числото n се съдържа цифра p , да се изведе на монитора “da”, в противен случай „ne”**

**#include <iostream>**

**using namespace std;**

**int main() {**

**unsigned long int n,p,k;**

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

**cin>>n;**

**cout<<"p=";**

**cin>>p;**

**do**

**{**

**k=n%10;**

**n=n/10;**

**}**

**while((k!=p)&&(n!=0));**

**if(k==p)cout<<"da"<<endl;**

**else cout<<"ne"<<endl;**

**return 0;**

**}**

**Изход:**

n=74892

p=9

da

**Пример 8:** Програмата отпечатва Hello започвайки от 10 до 1

**#include<iostream>**

**#include <cstdio>**

**using namespace std;**

**main()**

**{**

**int i = 10;**

**do{**

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

**i = i -1;**

**}**

**while ( i > 0 );**

**}**

**Резултат:**

Hello 10

Hello 9

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

Hello 1

**Пример 9:** Фактът, че DO винаги изпълнява тялото на своя цикъл поне веднъж, го прави перфектен за проверка на вход на меню. Например в тази програма се изисква от потребителя да въвежда своя избор, до въвеждане на правилен отговор.

**#include<iostream>**

**#include <cstdio>**

**using namespace std;**

**int main(void)**

**{**

**int a, b;**

**char ch;**

**cout<<"Do you want to:"<<endl;**

**cout<<"Add, Subtract, Multiply, or Divide? ";**

**cout<<"Enter first letter: ";**

**cin>>ch;**

**cout<<"";**

**cout<<"Enter first number: ";**

**cin>>a;**

**cout<<"Enter second number: ";**

**cin>>b;**

**if(ch=='A') cout<<a+b;**

**if(ch=='S') cout<<a-b;**

**if(ch=='M') cout<<a\*b;**

**if(ch=='D' && b!=0) cout<<a/b;**

**return 0;**

**}**

**Резултат:**

Do you want to:

Add, Subtract, Multiply, or Divide?

Enter first letter: D

Enter first number: 8

Enter second number: 4

2

**Пример 10 :** Цикълът Do е особено полезен, когато вашата програма очаква да се случи някакво събитие. Например тази програма очаква да се въведе числото q. Забележете, че тя съсдържа едно извикване на getchar() по-малко от еквивалентната програма, описана в секцията за цикъла while

**#include <stdio.h>**

**int main(void)**

**{**

**char ch;**

**do {**

**ch=getchar();**

**}while(ch!='q');**

**return 0;**

**}**

**Резултат:**

r

t

q

**Пример 11. Напишете програма, която чете знаци от клавиатурата и изписва на екрана малките букви като главни. Спрете, когато бъде натиснат Enter.**

**#include<iostream>**

**#include <cstdio>**

**using namespace std;**

**int main(void)**

**{**

**char ch;**

**cout<<"Enter lowercase letters.";**

**cout<<"Press enter to Quit.\n";**

**do**

**{**

**cin>>ch;**

**if(ch!='\r')cout<<(char)(ch-32);**

**}while(ch!='\r');**

**return 0;**

**}**

**Резултат:**

Enter lowercase letters.Press enter to Quit.

yrtsg

YRTSG

**Основни алгоритми, реализиращи се с циклични алгоритми**

1. **Проверка за коректност на данните**

**В много от задачите се налага да се въвеждат данни, които отговарят на предварително зададени условия. Следващата задача показва коректно съставено условие за проверка дали въведено число е в даден интервал. За реализацията на задачата е подходящо да се използва цикъла while.**

**От клавиатурата се въвежда естествено число n в интервала 5<=n<=50. Ако числото не е в този интервал, да се изисква ново въвеждане**

**При коректно зададен вход за n да се изчисли и изведе квадратът му**

**Пример 12:**

**#include <iostream>**

**using namespace std;**

**int main() {**

**int n;**

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

**cin>>n;**

**while((n<5)||(n>50))**

**cout<<"nekorektni danni!";**

**cout<<"Vavedi otnovo n!" <<endl;**

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

**cin>>n;**

**cout<<n\*n<<endl;**

**return 0;**

**}**

1. **Намиране на максимална и минимална стойност на елемент**

**Тук може да се разглеждат 2 случая :**

1. **броят на числата в редицата е предварително известен**
2. **броят на числата от редицата не е известен**

**За решаването на задачите е удобно да се декларират 3 променливи – max/ за въвеждане на максимален елемент/ , min/ за въвеждане на минимален елемент/ и chislo /за въвеждане на числата от редицата/.**

**Пример 11: Да се напише програма, която въвежда от клавиатурата естествено число n и след него n на брой реални числа.**

**Да се изведат максималното и минималното от тях.**

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int n;**

**cin>>n;**

**double max,min,chislo;**

**cin>>chislo;**

**min=chislo; //min и max се инициализират с първата въведена ст-ст**

**max=chislo;**

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

**{**

**cin>>chislo;**

**if(chislo>max)max=chislo;**

**if(chislo<min)min=chislo;**

**}**

**cout<<"Min="<<min<<endl;**

**cout<<"Max="<<max<<endl;**

**return 0;**

**}**

**Изход:**

3

34 56 78

Min=34

Max=78

**Пример 13: Да се напише програма, която въвежда последователно от клавиатурата реални числа до момента, в който се въведе число 0. Да се изведат на монитора минималното и максималното число, без да се включва нулата.**

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int n;**

**cin>>n;**

**double max,min,chislo;**

**cin>>chislo;**

**min=chislo; //min и max се инициализират с първата въведена ст-ст**

**max=chislo;**

**while(chislo!=0)**

**{**

**if(max<chislo)max=chislo;**

**if(min>chislo)min=chislo;**

**cin>>chislo;**

**}**

**cout<<"Min="<<min<<endl;**

**cout<<"Max="<<max<<endl;**

**return 0;**

**}**

**Изход:**

5

8

7

2

0

Min=2

Max=8

1. **Намиране на сумата и произведението на редица от числа:**

**Пример 14:**

**Алгоритъмът е следния:**

* **Декларира се променлива sum, в която ще се натрупва сумата**
* **Променливата sum се инициализира с 0**
* **Въвежда се поредното число и стойността му се добавя към тази на sum**
* **Стъпка 3 се извършва n пъти, колкото е броят на числата**
* **Извежда се стойността на променливата sum, която е натрупаната сума**

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**double sum, pr,chislo;**

**sum=0;**

**pr=1;**

**int n;**

**cin>>n;**

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

**{**

**cin>>chislo;**

**sum=sum+chislo;**

**pr=pr\*chislo;**

**}**

**cout<<"Sum="<<sum<<endl;**

**cout<<"Pr="<<pr<<endl;**

**return 0;**

**}**

**Изход:**

3

5 9 12

Sum=26

Pr=540

1. **Оператор BREAK**

**Конструкцията BREAK дава възможност за излизане от цикъл в която и да е точка на неговото тяло, като по този начин се заобикаля израза за нормално прекъсване**.

Ако в един цикъл се срещне конструкцията break , той незабавно се прекратява и програмата продължава с конструкцията след цикъла.

* **Операторът break прекратява изпълнението на съдържащия го оператор switch**
* **Оператора break прекратява изпълнението на оператор за цикъл, който го съдържа**

**Пример 15:** Този цикъл отпечатва само числата от 1 до 10

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int i;**

**for(i=1; i<100; i++) {**

**cout<<i;**

**if(i==10) break; /\* прекъсва брояча\*/**

**}**

**return 0;**

**}**

**Резултат:**

1 2 3 4 5 6 7 8 9 10

Конструкцията break може да бъде използвана във всеки от трите цикъла на С.

В един цикъл можете да поставите толкова конструкции break, колкото желаете. Тъй като прекалено много изходни точки от един цикъл могат да деструктират кода, по-добре е да използвате break само за специални цели, а не като нормален изход от цикъла.

**Пример 16: Тази програма изкарва числата, които се делят на 6 от 1 до 9999 като пита постоянно потребителят дали иска да изписва повече числа. Докато потребителят пише символа Y числата се показват, при натискане на символа N показването спира**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int i;**

**char ch;**

**/\* показва всички числа, които се делят на 6 \*/**

**for(i=1; i<10000; i++) {**

**if(!(i%6)) {**

**cout<<i;**

**cin>>ch;**

**if(ch=='N') break; /\* спира цикъла \*/**

**cout<<endl;**

**}**

**}**

**return 0;**

**}**

**Резултат:**

6, more? (Y/N)Y

12, more? (Y/N)

18, more? (Y/N)Y

24, more? (Y/N)

30, more? (Y/N)N

**Пример 17:** Дадена конструкция break ще предизвика прекратяване само на най-вътрешния цикъл. Тази програма отпечатва 5 пъти числата от 0 до 5

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int i,j;**

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

**{**

**for(j=0; j<100; j++)**

**{**

**cout<<j;**

**if(j==5) break;**

**}**

**cout<<endl;**

**}**

**return 0;**

**}**

**Резултат:**

012345

012345

012345

012345

012345

Причината, поради която С включва конструкцията break е, за да даде възможност на вашите програми за по-голяма ефективност.

**Пример 18:**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int day;**

**cin>>day;**

**switch(day)**

**{**

**case 1: cout<<"Mon";break;**

**case 2: cout<<"Tu";break;**

**case 3: cout<<"Wed";break;**

**case 4: cout<<"Thu";break;**

**case 5: cout<<"Fri";break;**

**case 6: cout<<"Sat";break;**

**case 7: cout<<"Sun";break;**

**default : cout<<"There is not a such a day";**

**}**

**return 0;**

**}**

**Пример 19: Да се състави програма, която по зададен месец и година, въведени от клавиатурата, определя броя на дните в този месец.**

**Решение:**

**Когато месецът е февруари, трябва да се направи проверка дали годината е високосна. Една година е високосна, когато се дели на 4 без остатък. Специално правило се прилага, когато последните две цифри на годината са нули. Това допълнително условие годината да е високосна е да се дели на 400 без остатък, а това се изразява със следното уравнение:**

**(((year%4==0)&&(year%100!=0))||(year%400==0))**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**unsigned int month,year;**

**cin>>month>>year;**

**switch(month)**

**{**

**case 2:if (((year%4==0)&&(year%100!=0))||(year%400==0))**

**cout<<29<<endl;**

**else cout<<28<<endl;**

**case 1:**

**case 3:**

**case 5:**

**case 7:**

**case 8 :**

**case 10:**

**case 12: cout<<31<<endl;break;**

**case 4:**

**case 6:**

**case 9:**

**case 11:cout<<30<<endl;**

**default : cout<<"there is no a such month";**

**}**

**return 0;**

**}**

**Пример 20: Да се състави програма, която по въведена дата (ден, месец, година) определя и извежда следващата дата.**

1. **Използване на конструкцията CONTINUE**

**Конструкцията continue е един вид противоположна на break.**

**Тя предизвиква изпълнението на следващата интерация, като пропуска кода, намиращ се между нея и условието на цикъла**. Например тази програма никога не отпечатва нищо:

**Пример 18:**

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int x;**

**for(x=0; x<100; x++) {**

**continue;**

**cout<<x; //това никога не се изпълнява**

**}**

**return 0;**

**}**

При всяко достигане на конструкцията continue тя предизвиква повторението на цикъла, като пропуска конструкцията cout.

**Continue** се използва рядко, защото не се срещат често случаи за уместното му прилагане.

**Пример 19:** Програма, която изчислява текущата сума на числата, въведени от потребителя. Преди добавянето на числото към сумата, програмата проверява дали то е въведена правилно, като изисква от потребителя да го въведе отново. Ако двете числа не съвпадат, програматa използва continue, за да започне цикълът отначало.

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int total, i, j;**

**total = 0;**

**do {**

**cout<<"Enter next number (0 to stop): ";**

**cin>>i;**

**cout<<"Enter number again: ";**

**cin>>j;**

**if(i != j) {**

**cout<<"Mismatch”;**

**continue;**

**}**

**total = total + i;**

**} while(i);**

**cout<<"Total is"<<total;**

**return 0;**

**}**

**Резултат:**

Enter next number (0 to stop): 9 3

Enter number again: Mismatch

Enter next number (0 to stop): 5 5

Enter number again: Enter next number (0 to stop):

**Избор между алтернативи чрез конструкцията SWITCH**

**Конструкцията if е подходяща за избор между две алтернативи, но тя лесно се затруднява при появата на повече такива. Решението на езика С на този проблем е с конструкцията SWITCH**.

**Това е конструкцията на С за избор от множество възможности**. Тя се използва за избор на един от няколко възможни, алтернативни начина за изпълнение на програмата.

Ето как изглежда общата форма на конструкцията switch:

**switch(стойност){**

**case константа1;**

**поредица от конструкции**

**break;**

**case константа2;**

**поредица от конструкции**

**break;**

**case константа3;**

**поредица от конструкции**

**break;**

**………………….**

**default;**

**поредица от конструкции**

**break;**

**}**

Поредицата от конструкции default се изпълнява, ако не се намери съвпадение. Default не е задължителна част от switch.

Ако всички сравнения пропаднат и няма default, тогава не се извършва никакво действие. Ако се намери съвпадение, конструкциите, асоциирани с този case се изпълняват до срещане на break, или, в случая с default или последния case, до стигане до края на switch.

**Пример 20:** Тази програма разпознава цифрите 1,2,3 и 4 и отпечатва името на въведената.

**#include <iostream>**

**using namespace std;**

**int main() {**

**int i;**

**cout<<"Enter a number between 1 and 4: ";**

**cin>>i;**

**switch(i) {**

**case 1:**

**cout<<"one";**

**break;**

**case 2:**

**cout<<"two";**

**break;**

**case 3:**

**cout<<"three";**

**break;**

**case 4:**

**cout<<"four";**

**break;**

**default:**

**cout<<"Unrecognized Number";**

**}**

**return 0;**

**}**

**Резултат:**

Enter a number between 1 and 4: 3

Three

Конструкцията switch се различава от if по това, че може да проверява само за равенство, докато изразът за проверка на if може да е от всякакъв тип.

Също така, switch работи само с типовете int и char.

ANSI С разрешава поне 257 конструкции case. Поради причини, свързани с ефективността, трябва да ограничавате броя на case конструкциите до много по-малък брой. Също така, в един и същ switch не могат да съществуват два case константи с идентични стойности.

Възможно е един switch да бъде част от поредица конструкции на по-външен switch. Това се нарича **вложен switch.** Ако case константите на вътрешния и на външния switch съдържат общи стойности, това няма да предизвика конфликт.

**switch(a) {**

**case 1:**

**switch(b) {**

**case 0: printf("b is false");**

**break;**

**case 1: printf("b is true");**

**}**

**break;**

**case 2:**

**.**

**.**

**.**

Всеки компилатор по ANSI стандарта ще позволи влагането на switch конструкции на поне 15 нива.

**Пример 21:**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int a, b;**

**char ch;**

**cout<<"Do you want to:";**

**cout<<"Add, Subtract, Multiply, or Divide?";**

**/\* force user to enter a valid response \*/**

**do {**

**cout<<"Enter first letter: ";**

**cin>>ch;**

**} while(ch!='A' && ch!='S' && ch!='M' && ch!='D');**

**cout<<" ";**

**cout<<"Enter first number: ";**

**cin>>a;**

**cout<<"Enter second number: ";**

**cin>>b;**

**switch(ch) {**

**case 'A': cout<< a+b;**

**break;**

**case 'S': cout<<a-b;**

**break;**

**case 'M': cout<< a\*b;**

**break;**

**case 'D': if(b!=0) cout<<a/b;**

**}**

**return 0;**

**}**

**Резултат:**

Enter first letter: A

Enter first number: 3

Enter second number: 5

8

**Пример 22:** Ако липсва конструкция break, изпълнението на програмата „попада” на следващия case и спира едва когато срещне break или стигне края на switch

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**char ch;**

**do {**

**cout<<"Enter a character, q to quit: ";**

**cin>>ch;**

**cout<<endl;**

**switch(ch) {**

**case 'a':**

**cout<<"Now is "<<endl;**

**case 'b':**

**cout<<"the time "<<endl;**

**case 'c':**

**cout<<"for all good men"<<endl;**

**break;**

**case 'd':**

**cout<<"The summer "<<endl;**

**case 'e':**

**cout<<"soldier "<<endl;**

**}**

**} while(ch != 'q');**

**return 0;**

**}**

**Резултат:**

Enter a character, q to quit: a

Now is the time for all good men

Enter a character, q to quit:

**Пример 23:** Тази програма разделя буквите на гласни и съгласни. Две или повече конструкции case поделят една и съща поредица от конструкции, без да се налага дублиране на кода.

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**char ch;**

**cout<<"Enter the letter: ";**

**cin>>ch;**

**switch(ch) {**

**case 'a':**

**case 'e':**

**case 'i':**

**case 'o':**

**case 'u':**

**case 'y':**

**cout<<" is a vowel\n"; //гласна**

**break;**

**default:**

**cout<<" is a consonant"; //съгласна**

**}**

**return 0;**

**}**

**Резултат:**

Enter the letter: o

is a vowel

**Зад. Въвеждаме дата и програмата извежда кои ден от седмицата е**

**#include <iostream>**

**using namespace std;**

**int main()**

**{int x,den,mes,god,visok;**

**cin>>den>>mes>>god;**

**x=god%100;**

**x=x+x/4;**

**x=x+den;**

**if(((god%4==0)&&(god%100!=0))||(god%400==0))**

**visok=-1;**

**else visok=0;**

**switch(mes)**

**{**

**case 1:x=x+1+visok;break;**

**case 2:x=x+4+visok;break;**

**case 3:case 11:x=x+4;break;**

**case 5:x=x+2;break;**

**case 6:x=x+5;break;**

**case 8:x=x+3;break;**

**case 9:x=x+6;break;**

**case 10:x=x+1;break;**

**}**

**if(god<1900) x=x+2;**

**else if(god<200) x=x+6;**

**else x=x+5;**

**switch(x%7)**

**{**

**case 1:cout<<"Ponedelnik"<<endl;break;**

**case 2:cout<<"Vtornik"<<endl;break;**

**case 3:cout<<"Sriada"<<endl;break;**

**case 4:cout<<"Chetvurtuk"<<endl;break;**

**case 5:cout<<"Petuk"<<endl;break;**

**case 6:cout<<"Subota"<<endl;break;**

**case 0:cout<<"Nedelia"<<endl;break;**

**}**

**system("PAUSE");**

**return 0;**

**}**

**Изход:**

**1.14.2014**

**Vtornik**

**КОНСТРУКЦИЯТА GOTO**

GOTO е конструкция за безусловен преход. Тя не се използва много често, тъй като тя деструктурира програмата и ако се използва често, може да направи програмата практически неразбираема. Освен това не съществува конструкция, която изисква GOTO.

GOTO може да извършва преход само в текущата функция. Тя не може да прескача между различни функции.

GOTO работи с помощта на етикет / валидно име на идентификатор, следван от двоеточие/

**Пример :** В тази програма конструкцията goto прескача конструкцията printf()

**goto mylabel;**

**cout<<"This will not print.";**

**mylabel: cout<<"This will print.";**

**Пример 24: Тази програма използва goto, за да създаде еквивалент на цикъл for, изпълняван от 1 до 10**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int i;**

**i = 1;**

**again:**

**cout<< i;**

**i++;**

**if(i<10) goto again;**

**return 0;**

**}**

**Резултат:**

1 2 3 4 5 6 7 8 9

**ДОПЪЛНИТЕЛНИ ЗАДАЧИ**

**Зад 1. Да се състави програма, която въвежда от клавиатурата естествено число n и n на брой символи. Програмата да извежда броя на тези от въведените символи, които са малки латински букви.**

***За да се реши задачата трябва да се провери дали всеки въведен сивол е малка латинска буква.***

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**unsigned int n,br=0;**

**char ch;**

**cin>>n; //брой на въвежданите числа**

**for(int i=0;i<n;i++)**

**{**

**cin>>ch; //въвеждаме символи**

**if(ch>='a'&&ch<='z')br++;**

**}**

**cout<<br<<endl;**

**return 0;**

**}**

**Зад.2 Да се състави програма, която въвежда от клавиатурата цифра k, естествено число n и n на брой естествени числа. Програмата да извежда броя на онези числа от въведените, които са k-цифрени.**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**unsigned int k,n,chislo,br\_c,br=0;**

**cin>>k>>n;**

**for(int i=0;i<n;i++)**

**{**

**br\_c=0;**

**cin>>chislo;**

**while(chislo)**

**{**

**br\_c++;**

**chislo/=10;**

**}**

**if(br\_c==k)br++;**

**}**

**cout<<br<<endl;**

**return 0;**

**}**

**Зад. 3 да се състави програма, която въвежда от клавиатурата цели положителни числа. За край на въвеждането служи числото нула. Програмата да намира и извежда сумата на четните и броя на нечетните от въведените числа.**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**unsigned int chislo,sum=0,br=0;**

**cin>>chislo;**

**while(chislo)**

**{**

**if(chislo%2==0)sum+=chislo;**

**else br++;**

**cin>>chislo;**

**}**

**cout<<sum<<" "<<br<<endl;**

**return 0;**

**}**

**Зад. 4 Да се състави програма, която въвежда от клавиатурата цели положителни числа. За край на въвежданото число служи числото нула. Програмата да намира и извежда най-голямото от тях, което е нечетно число. В случай , че въведените числа са само четни, да се изведе подходящо съобщение.**

***При решаване на тази задача се инициализира променливата max , в която ще се запази най-голямото число.***

***Тъй като по условие ще се въвеждат само положителни числа, max се инициализира с минимална стойност 0. В края на програмата се прави проверка дали max е 0 или различна от 0. Така става ясно дали са били въведени нечетни числа.***

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**unsigned int chislo,max=0;**

**cin>>chislo;**

**while(chislo)**

**{**

**if(chislo%2==1&&max<chislo) max=chislo;**

**cin>>chislo;**

**}**

**if(max)cout<<max<<endl;**

**else cout<<"ne ste vaveli nechetni chisla"<<endl;**

**return 0;**

**}**

**Зад. 5 Върху числовата ос до 100 см са нанесени деления през 1 см. Да се състави програма, която въвежда от клавиатурата естествено число n (2<=n<=100) и n на брой двойка цели числа, които задават краищата на отсечки върху числовата ос. Програмата да извежда дължината на най-голямата и най-малката зададена отсечка.**

***В условието на задачата не е конкретизирано кой край на отсечката се задава като първи( дали в по-близкия или в по-далечния край на цифровата ос). Нека променливите а1 и а2 служат за въвеждане двойката числа, указващи краищата на отсечката. За да се намери дължината на отсечката, при всяка двойка числа, от по-голямото се вади по-малкото и се помни в променливата а. Спрямо намерената дължина се прилага алгоритъм за намиране на максимална и минимална стойност.***

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**unsigned int a,a1,a2,max=0,min=100,n;**

**cin>>n;**

**for(int i=0;i<n;i++)**

**{**

**cin>>a1>>a2;**

**if(a1>a2) a=a1-a2;**

**else a=a2-a1;**

**if(max<a) max=a;**

**if(min>a) min=a;**

**}**

**cout<<max<<" "<<min<<endl;**

**return 0;**

**}**

**Зад. 6 Да се състави програма, която въвежда от клавиатурата цяло число n и след него брой цели числа. Програмата да проверява има ли измежду тях последователни равни числа и извежда съобщение Yes или No в съответния случай.**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int n,k,chislo,flag=0;**

**cin>>n;**

**cin>>k;**

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

**{**

**cin>>chislo;**

**if(chislo==k)**

**{**

**flag=1;**

**break;**

**}**

**k=chislo;**

**}**

**if(flag)cout<<"yes"<<endl;**

**else cout<<"no"<<endl;**

**return 0;**

**}**

**Зад. 7 Да се състави програма, която въвежда от клавиатурата цяло число n (100<n<1000000). Програмата да намира и извежда броя на цифрите на числото n , които са четни**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**long n,br=0;**

**cin>>n;**

**while(n)**

**{**

**if((n%10)%2==0)br++;**

**n/=10;**

**}**

**cout<<br<<endl;**

**return 0;**

**}**

**Зад. 8 Да се състави програма, която въвежда от клавиатурата цяло число n (0<n<1000000) и извежда броя на единиците в двоичния му код.**

***Алгоритъм:***

1. ***Докато числото има цифри, се намира остатъкът при целочислено деление на 2 и се прибавя към променлива брояч.***
2. ***Числото n получава нова стойност, която е частното при целочисленото му деление на 2***
3. ***Този процес продължава, докато частното при деление на 2 стане 0***

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int n,br=0;**

**cin>>n;**

**do**

**br+=n%2;**

**while(n/=2);**

**cout<<br<<endl;**

**return 0;**

**}**

**или:**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int n,br=0;**

**cin>>n;**

**while(n)**

**{**

**br+=n%2;**

**n/=2;**

**}**

**cout<<br<<endl;**

**return 0;**

**}**

**Зад. 9 Да се състави програма, която въвежда от клавиатурата цяло число n (20<n<1000000). Програмата да намира и извежда средноаритмитичното от цифрите на числото n.**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

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

**cin>>n;**

**while(n)**

**{**

**sum+=n%10;**

**if(max<n%10)max=n%10;**

**n/=10;**

**}**

**cout<<sum<<" "<<max<<endl;**

**return 0;**

**}**

**Зад. 10 Да се състави програма, която въвежда от клавиатурата цяло число n (0<n<1000000). Програмата да извежда двоичния му код.**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

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

**cin>>n;**

**while(n)**

**{**

**sum+=n%10;**

**br++;**

**n/=10;**

**}**

**cout<<(double)sum/br<<endl;**

**return 0;**

**}**

**Зад. 11 Редицата 1,2,3,5,8,13,21,......., където всяко следващо число е сума от предходните две, се нарича редица на Фибоначи. Съставете програма, която въвежда от клавиатурата число n и намира n-тото число на Фибоначи.**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int a1=1,a2=1,an,n,i=3;**

**cin>>n;**

**an=0;**

**if(n==1||n==2) an=1;**

**else**

**while(i<=n)**

**{**

**an=a1+a2;**

**a1=a2;**

**a2=an;**

**i++;**

**}**

**cout<<an<<endl;**

**return 0;**

**}**

**Зад. 12 Съставете програма, която въвежда от клавиатурата n и намира най-близкото число от редицата на Фибоначи, което е по-голямо от числото n**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int n,f1=1,f2=1;**

**cin>>n;**

**while(f2<=n)**

**{**

**f2+=f1;**

**f1=f2-f1;**

**}**

**cout<<f2<<endl;**

**return 0;**

**}**

**Зад. 13 Да се състави програма, която въвежда от клавиатурата целите числа p и q (p<q). Програмата да извежда простите числа в интервала [p,q].**

**#include<iostream>**

**using namespace std;**

**int main(void)**

**{**

**int p,q,flag,del;**

**cin>>p>>q;**

**for(int i=p;i<=q;i++)**

**{**

**flag=1;**

**for(del=2;del<=i/2;del++)**

**if(i%del==0)**

**{**

**flag=0;**

**break;**

**}**

**if(flag&&i>1)cout<<i<<" ";**

**}**

**return 0;**

**}**

**Зад. 14 Да се състави програма, която въвежда от клавиатурата естествено число p и го извежда, разложено на прости множители.**

**#include<iostream>**

**#include <cstdio>**

**using namespace std;**

**int main(void)**

**{**

**int p,i=2;**

**cin>>p;**

**cout<<p<<" ";**

**if(p==1)cout<<1;**

**do**

**if(p%i==0)**

**{**

**p/=i;**

**if(p==1)cout<<i;**

**else cout<<i<<"\*";**

**}**

**else i++;**

**while(p>1);**

**cout<<endl;**

**return 0;**

**}**

**Зад. 15 Да се състави програма, която въвежда от клавиатурата цяло число n (0<n<1000) и извежда двоичния му код**

**#include <iostream>**

**using namespace std;**

**int main(void)**

**{**

**int n;**

**long bin=0,pow10=1;**

**cin>>n;**

**while(n)**

**{**

**bin+=n%2\*pow10;**

**pow10\*=10;**

**n/=2;**

**}**

**cout<<bin<<endl;**

**return 0;**

**}**