REPETITION STRUCTURE - EXERCISE

Submitted By-Sakshi
Roll No- 88001

Q.3.1.1 Differentiate between while and do-while.

Ans.

While	do while
1. Entry controlled loop.	1. Exit controlled loop.
2. It runs zero or more times.	2. It runs one or more times.
3. Variable in test condition must be initialized prior to entering loop.	3. Not necessary for variable in test condition to be initialized prior to entering loop.

4. while(condition)
{
//do something;
}
	4. do{
	//do something;
	}while(condition);

Q.3.1.2 Predict the output.

```
i) int no=1;
do{
cout<<2*no;
no=no+1;
}while(no<=3);
```

Output : 2 4 6

```
ii) int no=1;
   while(no<=3)
   {cout<<2*no;
    no=no+1;
    }
```

Output : 2 4 6

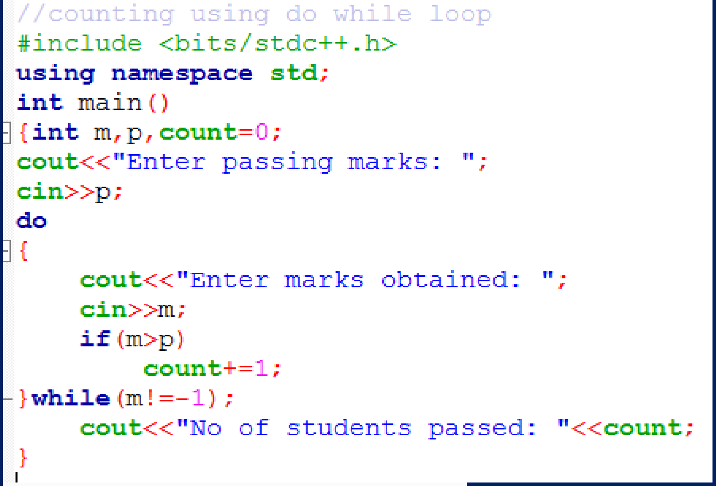
Q.3.1.3 The following program is supposed to enter input from user and displays them until 0 is entered, but a segment is missing. What is it?

```
cin>>no;
while(no!=0)
{
        cout<<no;
        cin>>no;    //this is the correction
}
```

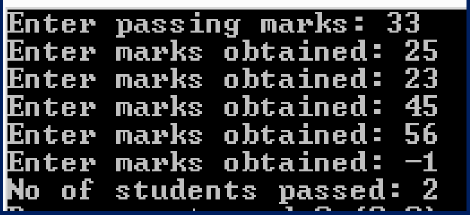
Ans. There must be a cin statement after cout like : cin>>no;

Q.3.1.4 Replace the while loop in the counting program and the summation program with do-while loop.

COUNTING PROGRAM:

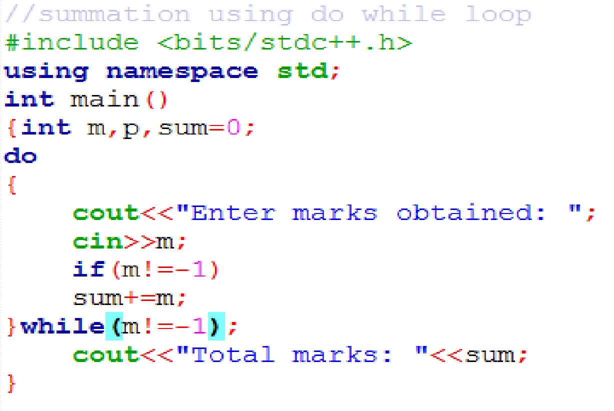


```
//counting using do while loop
#include <bits/stdc++.h>
using namespace std;
int main()
{int m,p,count=0;
cout<<"Enter passing marks: ";
cin>>p;
do
{
    cout<<"Enter marks obtained: ";
    cin>>m;
    if(m>p)
        count+=1;
}while(m!=-1);
    cout<<"No of students passed: "<<count;
}
```

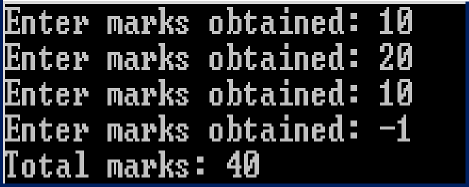


```
Enter passing marks: 33
Enter marks obtained: 25
Enter marks obtained: 23
Enter marks obtained: 45
Enter marks obtained: 56
Enter marks obtained: -1
No of students passed: 2
```

SUMMATION PROGRAM:

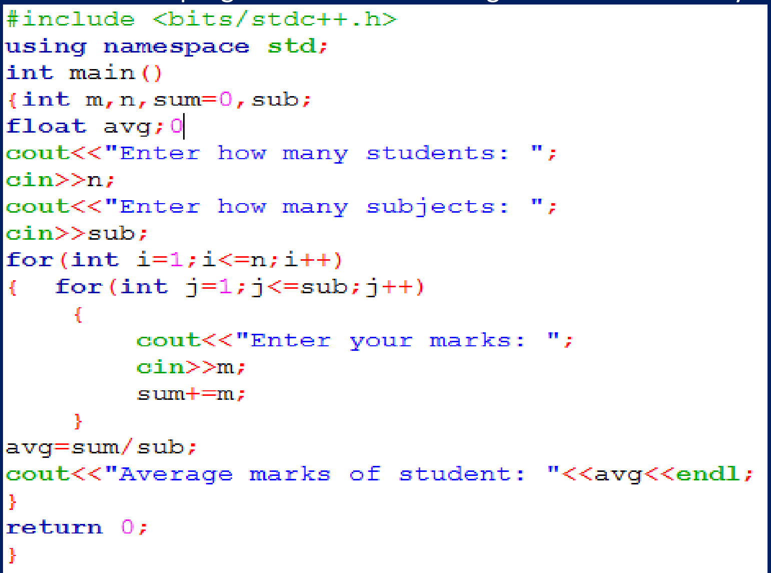


```
//summation using do while loop
#include <bits/stdc++.h>
using namespace std;
int main()
{int m,p,sum=0;
do
{
    cout<<"Enter marks obtained: ";
    cin>>m;
    if(m!=-1)
    sum+=m;
}while(m!=-1);
    cout<<"Total marks: "<<sum;
}
```

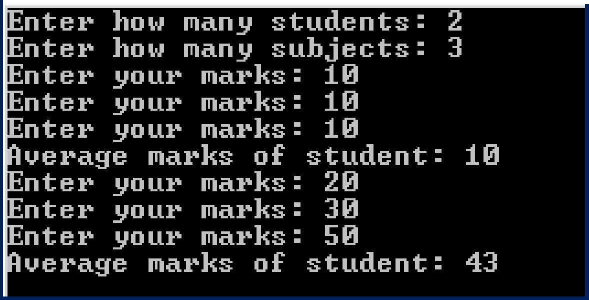


```
Enter marks obtained: 10
Enter marks obtained: 20
Enter marks obtained: 10
Enter marks obtained: -1
Total marks: 40
```

Q3.1.5 Write a program to find the average of marks scored by a set of students.



```
#include <bits/stdc++.h>
using namespace std;
int main()
{int m,n,sum=0,sub;
float avg;0
cout<<"Enter how many students: ";
cin>>n;
cout<<"Enter how many subjects: ";
cin>>sub;
for(int i=1;i<=n;i++)
{   for(int j=1;j<=sub;j++)
        {
            cout<<"Enter your marks: ";
            cin>>m;
            sum+=m;
        }
avg=sum/sub;
cout<<"Average marks of student: "<<avg<<endl;
}
return 0;
}
```



```
Enter how many students: 2
Enter how many subjects: 3
Enter your marks: 10
Enter your marks: 10
Enter your marks: 10
Average marks of student: 10
Enter your marks: 20
Enter your marks: 30
Enter your marks: 50
Average marks of student: 43
```

Q3.2.1 Define a counter controlled loop with example.

Ans. A counter controlled loop is executed a fixed number of times. A 'for' loop is an example.

For example:

```
{
for(int i=1;i<=4;i++)
        cout<<l;
return 0;
}
```

Q3.2.2 Predict the output:

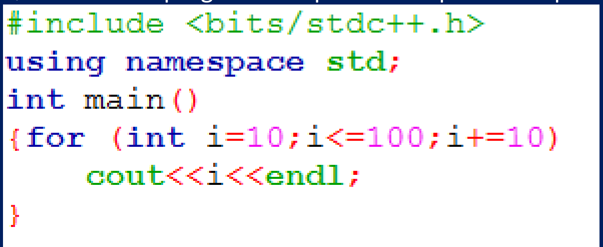
```
i)for(int count=10;count<6;count=count-2)
        cout<<count<<endl;
```

Output : No output

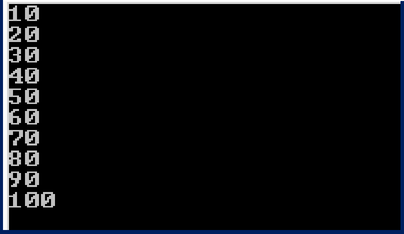
```
ii) )for(int count=0;count>6;count=count--)
        cout<<count<<endl;
```

Output : No output

Q3.2.3 Write a program that prints multiple of 10 upto 100.



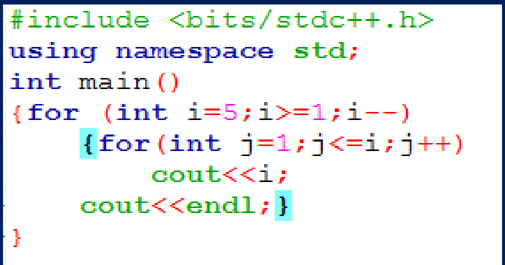
```
#include <bits/stdc++.h>
using namespace std;
int main()
{for (int i=10;i<=100;i+=10)
    cout<<i<<endl;
}
```



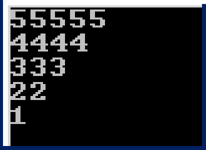
```
10
20
30
40
50
60
70
80
90
100
```

Q3.2.4 Write a program that prints pattern:

```
55555
4444
333
22
1
```



```
#include <bits/stdc++.h>
using namespace std;
int main()
{for (int i=5;i>=1;i--)
    {for(int j=1;j<=i;j++)
        cout<<i;
    cout<<endl;}
}
```



```
55555
4444
333
22
1
```

Q.3.3.1 What are the differences between break, return and continue?

Ans. All three are jump statements.

break	continue	return
Use to exit the loop before the exit conditions are met.	It keeps continuity in the loop without executing the statements written after the continue statement.	It will cause the function to return immediately to the calling function.

Q3.3.2 Predict the output:

```
i)for(i= 10 ; i>4 ; i--)
{
cout<<i<<endl;
if(i==4)
        break;
}
```

Output : 10

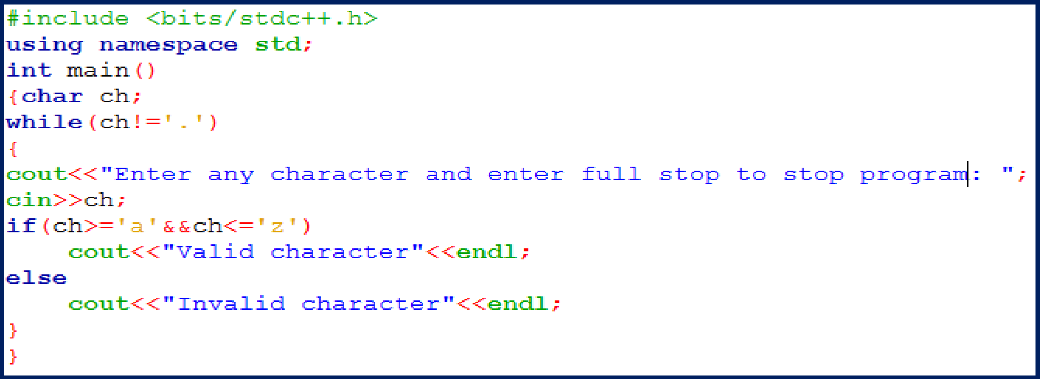
```
         9
         8
         7
         6
         5
```

```
ii) for(int n=10;n>0;n--)
{if(n==5)
continue;
cout<<n<<",";
}
cout<<"done"<<endl;
```

Output : 10,9,8,7,6,4,3,2,1,done

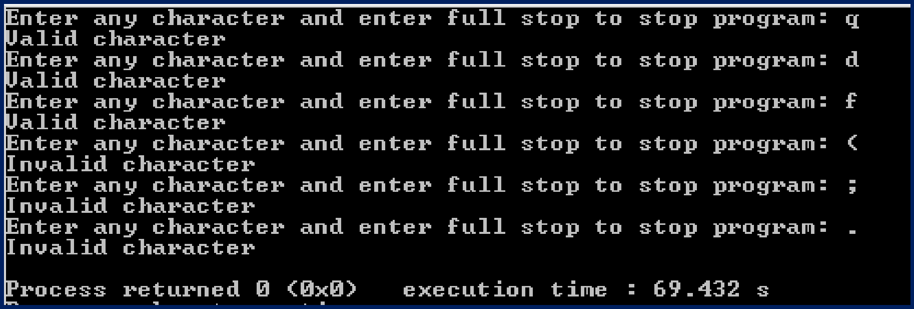
Q.3.3.3 Write a program that accepts a character input and validates that the character is only between a to z.

Program:



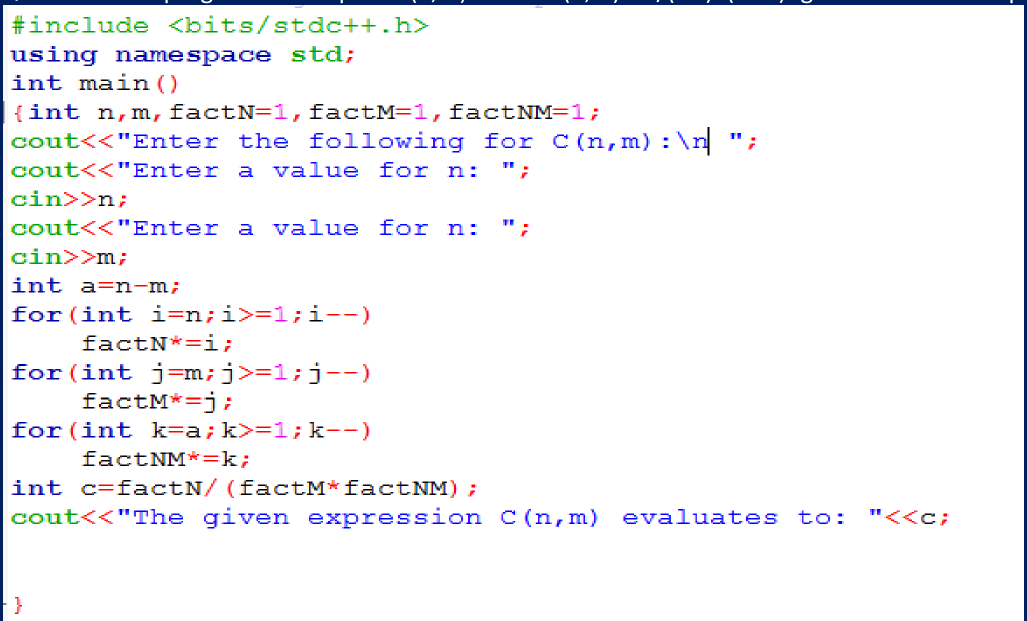
```
#include <bits/stdc++.h>
using namespace std;
int main()
{char ch;
while(ch!='.')
{
cout<<"Enter any character and enter full stop to stop program: ";
cin>>ch;
if(ch>='a'&&ch<='z')
    cout<<"Valid character"<<endl;
else
    cout<<"Invalid character"<<endl;
}
}
```

Output:

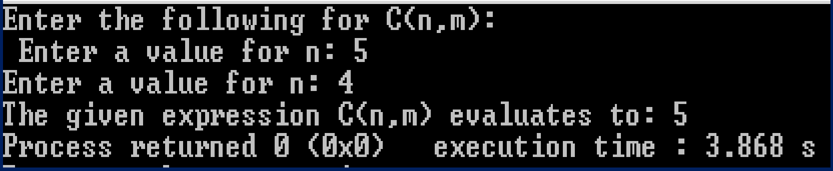


```
Enter any character and enter full stop to stop program: q
Valid character
Enter any character and enter full stop to stop program: d
Valid character
Enter any character and enter full stop to stop program: f
Valid character
Enter any character and enter full stop to stop program: (
Invalid character
Enter any character and enter full stop to stop program: ;
Invalid character
Enter any character and enter full stop to stop program: .
Invalid character

Process returned 0 (0x0)   execution time : 69.432 s
```

Q3.3.4 Write a program to compute C(n,m) where C(n,m)=n!/(m!)*(n-m)! given n and m as input.



```
#include <bits/stdc++.h>
using namespace std;
int main()
{int n,m,factN=1,factM=1,factNM=1;
cout<<"Enter the following for C(n,m):\n ";
cout<<"Enter a value for n: ";
cin>>n;
cout<<"Enter a value for n: ";
cin>>m;
int a=n-m;
for(int i=n;i>=1;i--)
    factN*=i;
for(int j=m;j>=1;j--)
    factM*=j;
for(int k=a;k>=1;k--)
    factNM*=k;
int c=factN/(factM*factNM);
cout<<"The given expression C(n,m) evaluates to: "<<c;


}
```



```
Enter the following for C(n,m):
 Enter a value for n: 5
Enter a value for n: 4
The given expression C(n,m) evaluates to: 5
Process returned 0 (0x0)   execution time : 3.868 s
```

===========+==================+======END=======+=============+=================

FUNCTIONS- EXERCISE

Submitted By-Sakshi

Roll No- 88001

Q1. What is the return type of the function with prototype: "int func(char a, float b, double c);"

A. char

B. int

C. float

D. double

Ans. (B) int

Q2. Which of the following is a valid function call (assuming the function exists)?

A. abc;

B. abc a, b;

C. abc();

D. int abc();

Ans. C) abc();

Q3. Which of the following is a complete function?

A. int qaz();

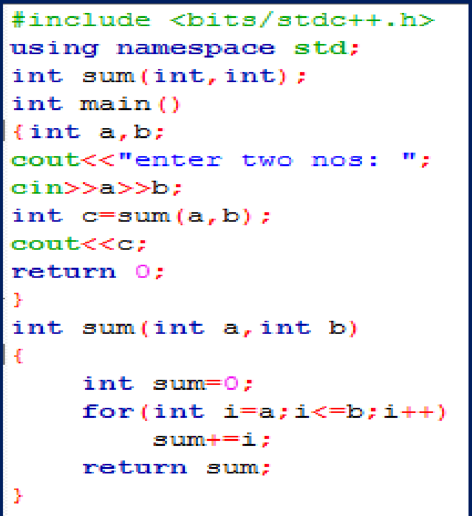
B. int qaz(int x) {return x=x+1;}

C. void qaz(int) {cout<<"Hello"}

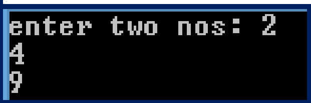
D. void qaz(x) {cout<<"Hello"}

Ans. (B) int qaz(int x) {return x=x+1;}

Q4. Write a C++ program which uses a function to find sum of numbers between two given numbers. These two numbers are passed to the function and the sum is returned from it.

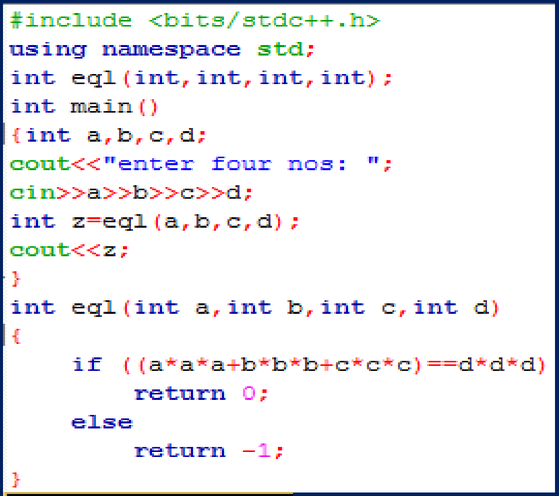


```
#include <bits/stdc++.h>
using namespace std;
int sum(int,int);
int main()
{int a,b;
cout<<"enter two nos: ";
cin>>a>>b;
int c=sum(a,b);
cout<<c;
return 0;
}
int sum(int a,int b)
{
     int sum=0;
     for(int i=a;i<=b;i++)
          sum+=i;
     return sum;
}
```

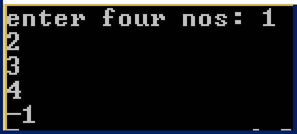


```
enter two nos: 2
4
9
```

Q5. Write a C++ program that invokes a function eql() to find whether four numbers a, b, c, d passed to eql() satisfy the equation a3 + b3 +c3 = d3 or not. eql() should return 0 if the above equation is satisfied with the passed numbers otherwise it returns -1.



```
#include <bits/stdc++.h>
using namespace std;
int eql(int,int,int,int);
int main()
{int a,b,c,d;
cout<<"enter four nos: ";
cin>>a>>b>>c>>d;
int z=eql(a,b,c,d);
cout<<z;
}
int eql(int a,int b,int c,int d)
{
     if ((a*a*a+b*b*b+c*c*c)==d*d*d)
          return 0;
     else
          return -1;
}
```



```
enter four nos: 1
2
3
4
-1
```

Q6. In what conditions will you use inline functions?

Ans. We use inline function when the function code is small and the function is to be expanded in line i.e compiler replaces the function call with function code.

Q7. How is an inline function different from a preprocessor macro?
Ans. Macro is an instruction which expands at the time of its invocation. Functions can also be defined, like macros. Similarly, the inline functions also expand at the point of its invocation. One primary difference between inline and macro function is that the inline functions are expanded during compilation, and the macros are expanded when the program is processed by the preprocessor.

Q8. Given the following code fragment:

```
int main()
{
float abc(float, float);
…
…
}
void qaz(void)
{
float x, y, s;
cin >>x;
cin >> y;
….
s = abc(x,y);
}
```

Will the above function work? Why?

Ans. No, it will not work.

1. Return type of function must be float at time of definition.
2. Function name is wrong.
3. Variable declaration inside the function declaration is incorrectly done- mismatched data type and less variables declared.

Q.9 When are the default arguments required in a function?
Ans. While calling a function if user don't want to enter the value of any variable then the default value is assigned.

Q.10 Can the effect of default arguments be achieved by overloading functions? How?
Ans. Yes, it can.

Q11. Write a declaration for a function called funct() that takes two arguments and returns a char value. The first argument is int and cannot be modified. The second argument is float with a default value of 3.14.
Ans. char funct(const int a, float b=3.14);

Q13. Identify the problem with the following code:

```
void large(int &i, int &j);
int main()
{
large (1,2);
}
void large(int &a, int &b)
{
if (a > b)
a = -1;
else
```

b= -1;
}

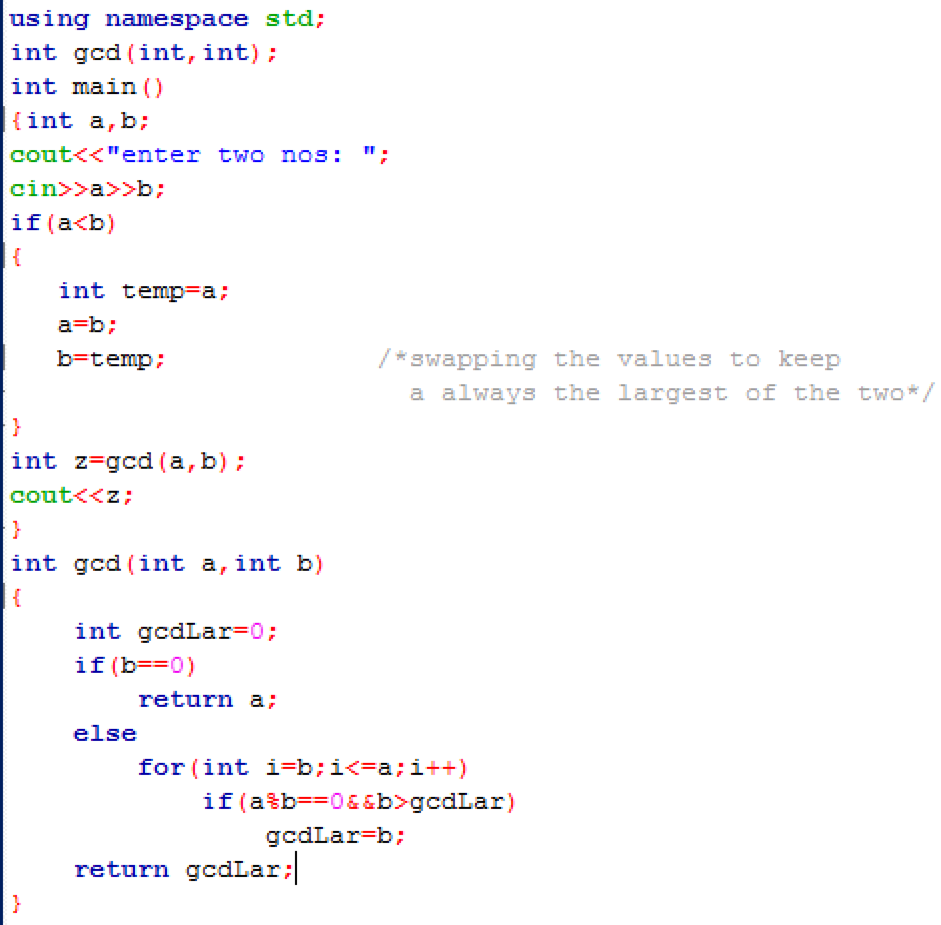
How can we correct it?

Ans. Here we have used ‘&’ which will store the value address in a and b instead of that we must have omitted ‘&’ so that only values are compared.

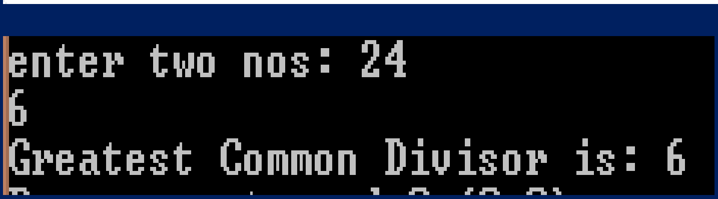
Q15. What is overloading of a function? When can it be useful?

Ans. Function overloading is the ability to create multiple functions of same name with different implementations. It is the most powerful weapon of C++and forms the basis of compile time polymorphism.

Q16. GCD function can be implemented in non recursive way using iterations. Write the function which calculates GCD in iterative way.



```
using namespace std;
int gcd(int,int);
int main()
{int a,b;
cout<<"enter two nos: ";
cin>>a>>b;
if(a<b)
{
   int temp=a;
   a=b;
   b=temp;             /*swapping the values to keep
                         a always the largest of the two*/
}
int z=gcd(a,b);
cout<<z;
}
int gcd(int a,int b)
{
    int gcdLar=0;
    if(b==0)
        return a;
    else
        for(int i=b;i<=a;i++)
            if(a%b==0&&b>gcdLar)
                gcdLar=b;
    return gcdLar;
}
```



```
enter two nos: 24
6
Greatest Common Divisor is: 6
```

============+============+========END============+===========+===============