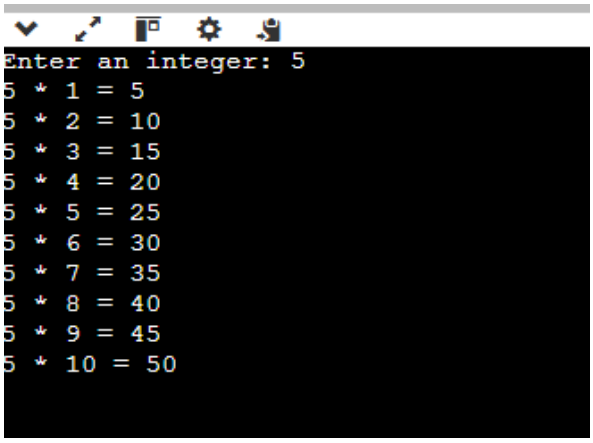
Questions C++

Q1 Print Multiplication Table of a Number

Ans

```
#include <iostream>

using namespace std;


void printTable(int num) {
    for(int i = 1; i <= 10; ++i) {
        cout << num << " * " << i << " = " << num * i << endl;
    }
}


int main() {
    int number;
    cout << "Enter an integer: ";
    cin >> number;
    printTable(number);


    return 0;
}
```

Output

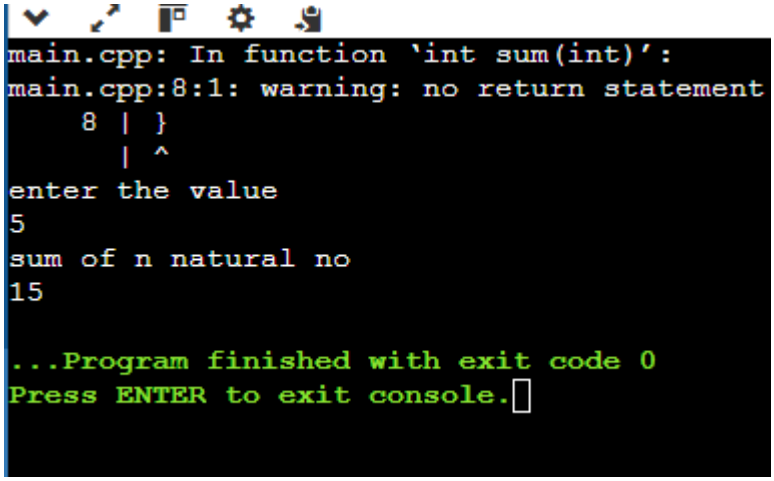


Q2. SUM OF ALL NATURAL NO

ANS

```
#include <iostream>
using namespace std;
int sum(int n)
{
   int s;
   s=n*(n+1)/2;
   cout<<s;
}
int main()
{
   int n;
   cout<<"enter the value"<<endl;
   cin>>n;
   cout<<"sum of n natural no"<<endl;
   sum(n);
}
```

OUTPUT:



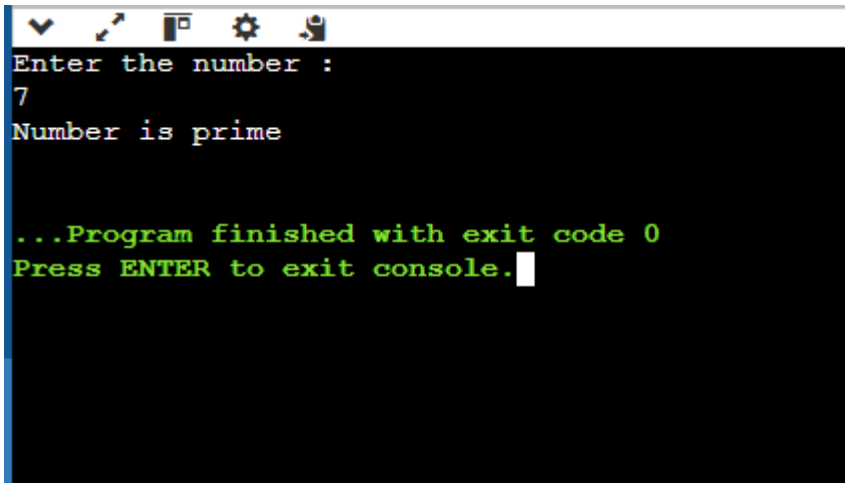
Q3 NUMBER IS PRIME OR NOT

ANS

```
#include <iostream>
using namespace std;
int main()
{
   int n,s;
   cout<<"Enter the number :"<<endl;
   cin>>n;
   s=0;
   for(int i=2;i<n;i++)
   {
      if(n%i==0)
      {
         s++;
      }
   }
   if(s>0)
   {
      cout<<"Number is not prime"<<endl;
   }
   else
```

```
    {
        cout<<"Number is prime"<<endl;
    }
}
```

OUTPUT:



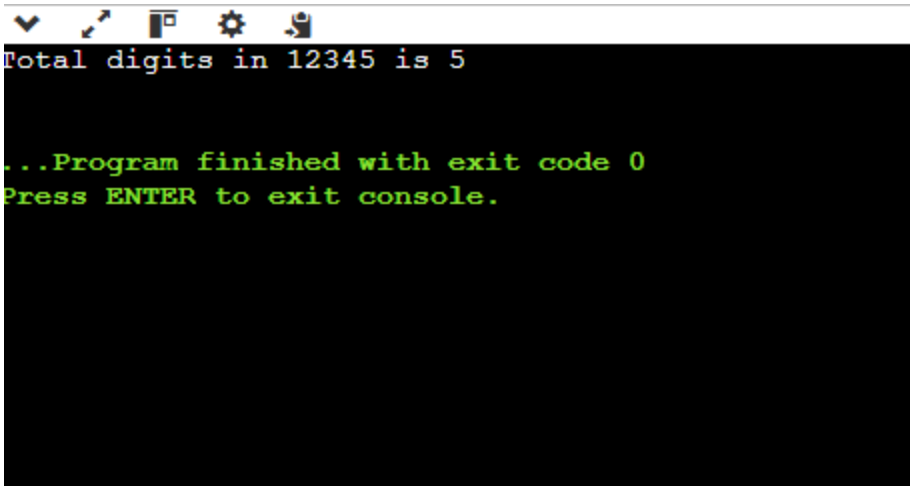
Q4. COUNT THE TOTAL NO OF DIGIT IN GIVEN NO N

ANS

```
#include <iostream>
using namespace std;

int countDigits(int n) {
    int count = 0;
    while (n > 0) {
        n /= 10;
        count++;
    }
    return count;
}

int main() {
    int n = 12345;
    cout << "Total digits in " << n << " is " << countDigits(n) << endl;
```

```
    return 0;
}
```

OUTPUT:

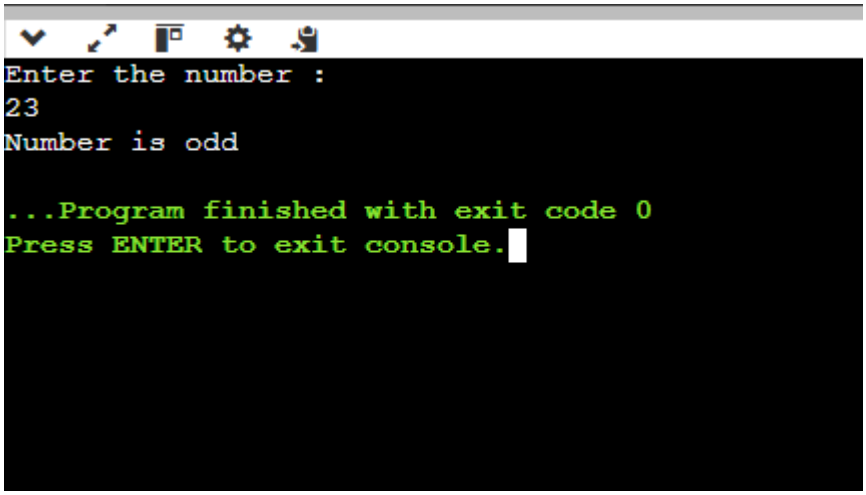


Q5 EVEN OR ODD

ANS

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cout<<"Enter the number :"<<endl;
    cin>>n;
    if(n%2==0)
    {
        cout<<"Number is even";
    }
    else
    {
        cout<<"Number is odd";
    }
}
```

OUTPUT:

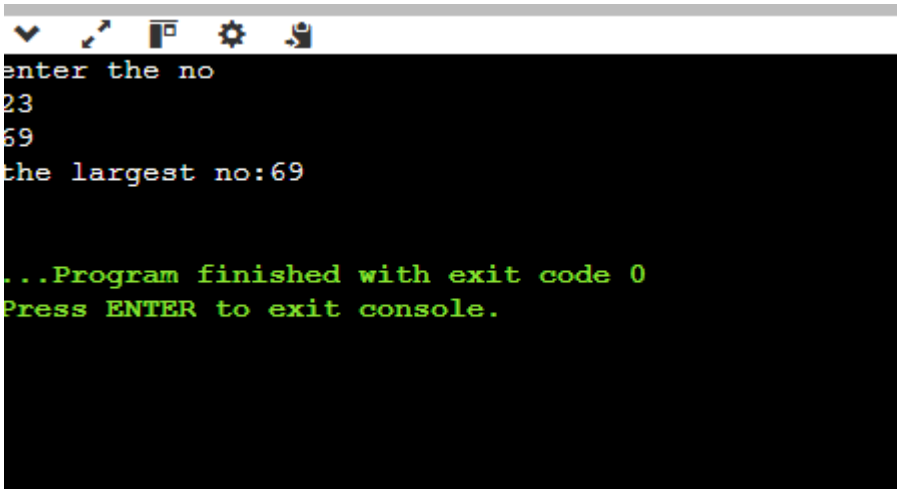


Q6 FIND THE LARGEST TWO NO

ANS

```
#include <iostream>
using namespace std;
int main()
{
   int a,b;
   cout<<"enter the no"<<endl;
   cin>>a>>b;
   if(a>b)
   {
      cout<<"the largest no:"<<a<<endl;
   }
   else
   {
      cout<<"the largest no:"<<b<<endl;
   }
}
```

OUTPUT

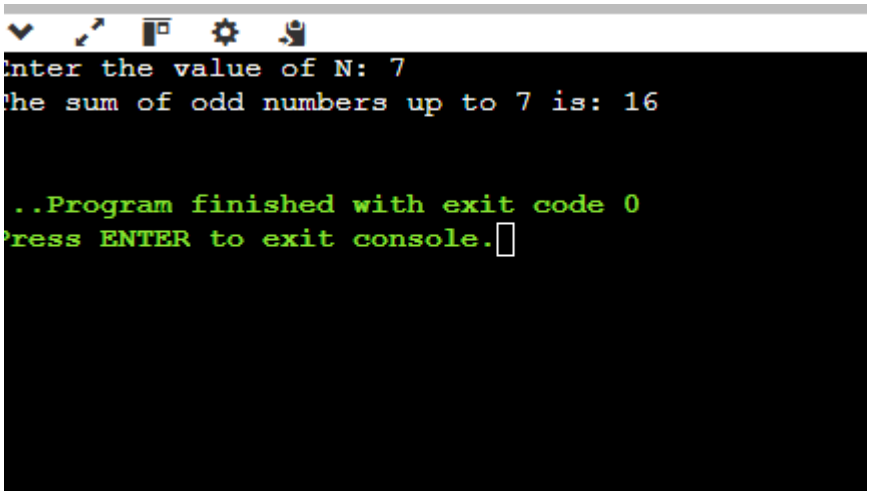


Q7 SUM OF ODD NO UPTO N

ANS

```
#include <iostream>

using namespace std;


int main() {

   int n, sum = 0;


   cout << "Enter the value of N: ";

   cin >> n;


   // Calculate the sum of odd numbers

   for (int i = 1; i <= n; i += 2) {

      sum += i;

   }


   cout << "The sum of odd numbers up to " << n << " is: " << sum << endl;

   return 0;

}
```

Output:



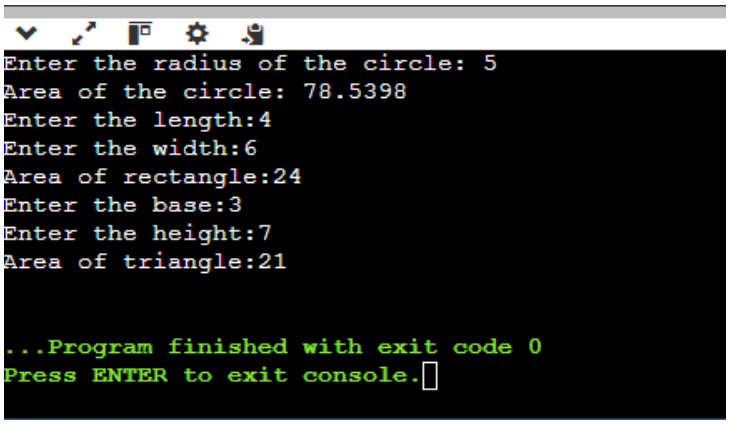
Q8. Write a program to calculate the area of different shapes using function overloading. Implement overloaded functions to compute the area of a circle, a rectangle, and a triangle.

Ans

```
#include <iostream>
#include <cmath>
using namespace std;

double area(double radius) {
    return M_PI * radius * radius;
}
double area(double length, double width){
    return length * width;
}
double area1(double base, double height){
    return 0.5 * base * height;
}

int main() {
    double radius, length, width, base, height;

    cout << "Enter the radius of the circle: ";
    cin >> radius;
```

```
    cout << "Area of the circle: " << area(radius) << endl;
    cout<<"Enter the length:";
    cin>> length;
    cout<<"Enter the width:";
    cin>> width;
    cout<<"Area of rectangle:"<< area(length,width) << endl;
    cout<<"Enter the base:";
    cin>> base;
    cout<<"Enter the height:";
    cin>> height;
    cout<<"Area of triangle:"<< area(base,height) << endl;


    return 0;
}
```

Output



Q9. Write a program that demonstrates function overloading to calculate the salary of employees at different levels in a company hierarchy. Implement overloaded functions to compute salary for:

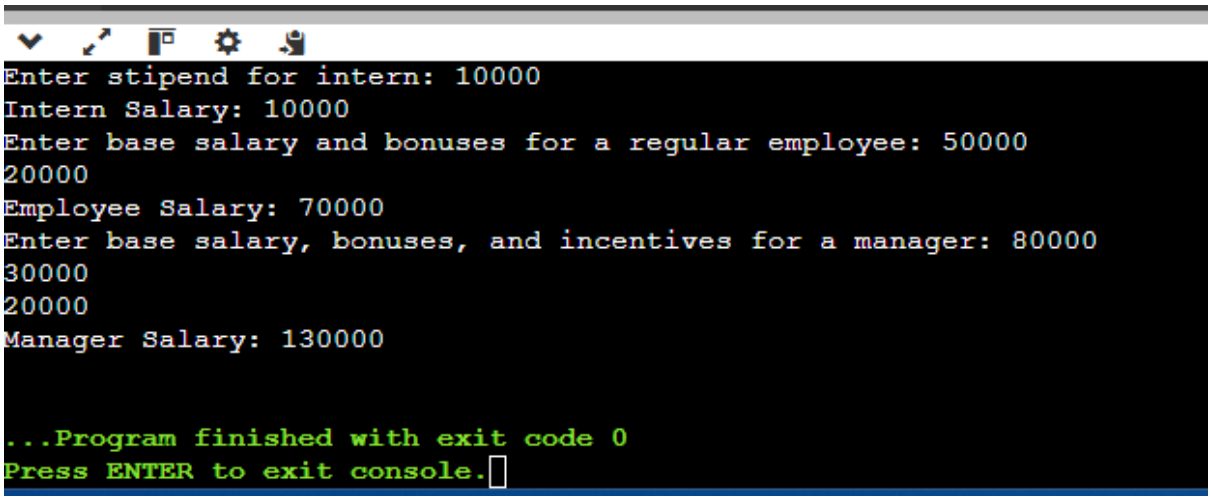
- Intern (basic stipend).
- Regular employee (base salary + bonuses).
- Manager (base salary + bonuses + performance incentives).

Ans

```
#include <iostream>
```

```
using namespace std;

int calculateSalary(int stipend) {
    return stipend;
}

int calculateSalary(int baseSalary, int bonuses) {
    return baseSalary + bonuses;
}

int calculateSalary(int baseSalary, int bonuses, int incentives) {
    return baseSalary + bonuses + incentives;
}

int main() {
    int stipend, baseSalary, bonuses, incentives;

    cout << "Enter stipend for intern: ";
    cin >> stipend;
    cout << "Intern Salary: " << calculateSalary(stipend) << endl;

    cout << "Enter base salary and bonuses for a regular employee: ";
    cin >> baseSalary >> bonuses;
    cout << "Employee Salary: " << calculateSalary(baseSalary, bonuses) << endl;

    cout << "Enter base salary, bonuses, and incentives for a manager: ";
    cin >> baseSalary >> bonuses >> incentives;
    cout << "Manager Salary: " << calculateSalary(baseSalary, bonuses, incentives) << endl;

    return 0;
}
```

Output



Q10 Create a C++ program that uses polymorphism to calculate the area of various shapes. Define a base class Shape with a virtual method calculateArea(). Extend this base class into the following derived classes:

Rectangle: Calculates the area based on length and width.

Circle: Calculates the area based on the radius.

Triangle: Calculates the area using base and height.

The program should use dynamic polymorphism to handle these shapes and display the area of each.

Ans

```
#include <iostream>

#include <cmath>


using namespace std;



class Shape {

public:

   virtual void calculateArea() = 0;

};


class Rectangle : public Shape {
```

```
private:
    float length, width;
public:
    Rectangle(float l, float w) : length(l), width(w) {}
    void calculateArea() override {
        cout << "Shape: Rectangle" << endl;
        cout << "Area: " << length * width << endl;
    }
};

class Circle : public Shape {
private:
    float radius;
public:
    Circle(float r) : radius(r) {}
    void calculateArea() override {
        cout << "Shape: Circle" << endl;
        cout << "Area: " << M_PI * radius * radius << endl;
    }
};

class Triangle : public Shape {
private:
    float base, height;
public:
    Triangle(float b, float h) : base(b), height(h) {}
    void calculateArea() override {
        cout << "Shape: Triangle" << endl;
        cout << "Area: " << 0.5 * base * height << endl;
    }
};
```

```
int main() {
    int shapeType;
    cout << "Enter shape type (1 for Rectangle, 2 for Circle, 3 for Triangle): ";
    cin >> shapeType;

    Shape* shape = nullptr;

    switch(shapeType) {
        case 1: {
            float length, width;
            cout << "Enter length and width of the rectangle: ";
            cin >> length >> width;
            shape = new Rectangle(length, width);
            break;
        }
        case 2: {
            float radius;
            cout << "Enter radius of the circle: ";
            cin >> radius;
            shape = new Circle(radius);
            break;
        }
        case 3: {
            float base, height;
            cout << "Enter base and height of the triangle: ";
            cin >> base >> height;
            shape = new Triangle(base, height);
            break;
        }
        default:
```

```
        cout << "Invalid shape type." << endl;
        return 1;
    }

    if(shape) {
        shape->calculateArea();
        delete shape;
    }

    return 0;
}
```

Output