DOMAIN WINTER CAMP

(Department of Computer Science and Engineering)

Name: Divya Malhotra UID: 22BCS10517 Section: 22BCS-KPIT 901

DAY 1

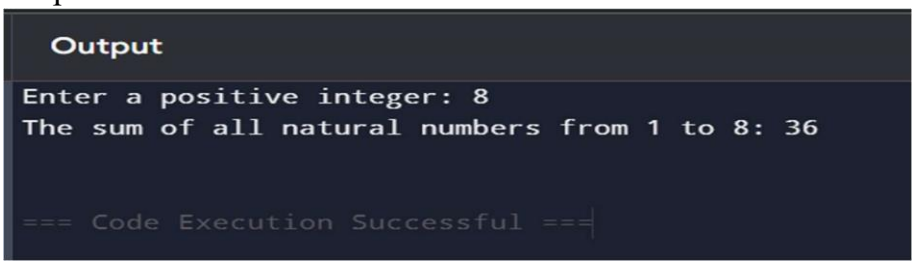
Ques 1. Calculate the sum of all natural numbers from 1 to n, where n is a positive integer. Use the formula:
Sum=n×(n+1)/2.
Take n as input and output the sum of natural numbers from 1 to n.

Program code:

```
#include <iostream>
using namespace std;
int sumOfNaturalNumbers(int n) {
    return n * (n + 1) / 2;
}
int main() {
    int n;
    cout << "Enter a positive integer: ";
cin >> n;
 if (n > 0) {
 int result = sumOfNaturalNumbers(n);
cout << "The sum of all natural numbers from 1 to " << n << ": " << result <<
endl;
}
```

```
else {        cout << "Please enter a positive integer." << endl;
}
return 0;}
```

Output:



Ques 2. Print all odd numbers between 1 and n, inclusive. Odd numbers are integers that are not divisible by 2. These numbers should be printed in ascending order, separated by spaces.

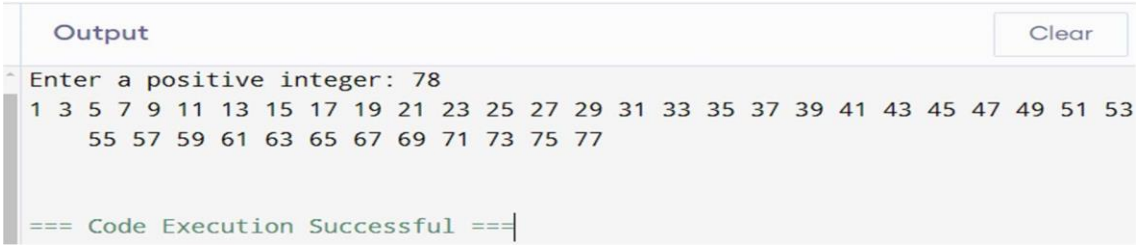
This problem is a simple introduction to loops and conditional checks. The goal is to use a loop to iterate over the numbers and check if they are odd using the condition i%2≠0.

Program code:

```
#include <iostream>
using namespace std;
int main() {
   int n;
   cout << "Enter a positive integer: ";
cin >> n;
if (n > 0) {
for (int i = 1; i <= n; i++) {
if (i % 2 != 0) {
cout << i << " ";
          }
}
```

```
 cout << endl;     } else {         cout << "Please enter a
positive integer." << endl;
}
return 0;
}
```

Output:



Ques 3. Count the total number of digits in a given number n. The number can be a positive integer. For example, for the number 12345, the count of digits is 5. For a number like 900000, the count of digits is 6.

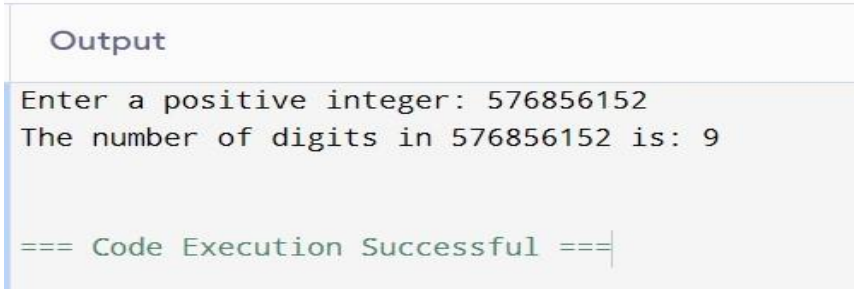
Given an integer n, your task is to determine how many digits are present in n. This task will help you practice working with loops, number manipulation, and conditional logic.

Program Code:

```
#include <iostream>
using namespace std;
int countDigits(int n) {
int count = 0;     while (n > 0) {         n /=
10; // Remove the last digit of n
count++; // Increment the digit count
    }
    return count;
}
int main() {
```

```
    int n;
    cout << "Enter a positive integer: ";
cin >> n;
if (n <= 0) {
        cout << "Please enter a positive integer." << endl;
    }
else {
 int result = countDigits(n);
cout << "The number of digits in " << n << " is: " << result << endl;
}
return 0;
}
```

Output:



Ques 4. 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.

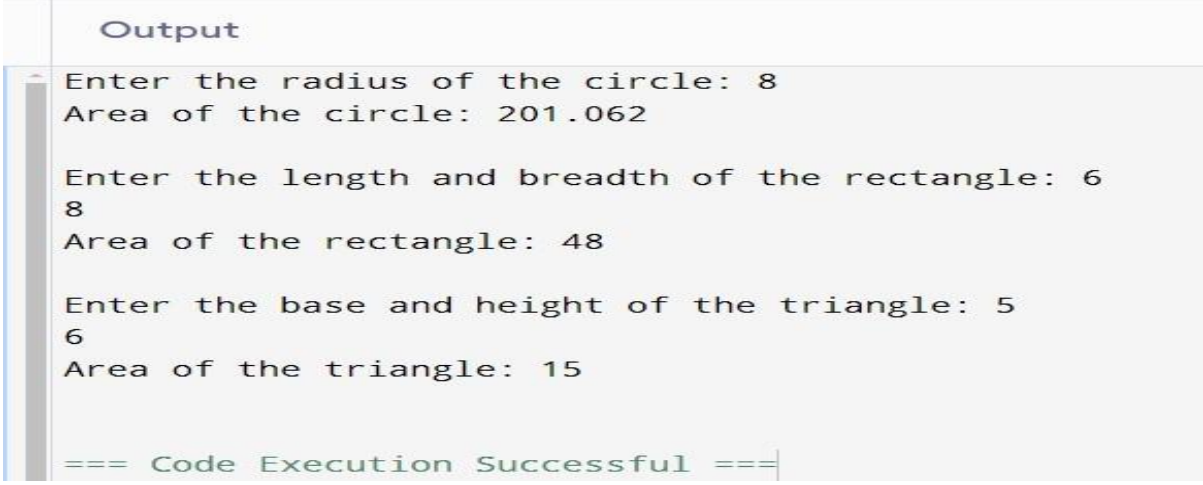
Program Code:

```
#include <iostream>
#include <cmath> using
namespace std; class
ShapeArea { public:
```

```
    double area(double radius) {
return M_PI * radius * radius;
}
double area(double length, double breadth) {
return length * breadth;
}
double area(double base, double height, bool isTriangle)
{
return 0.5 * base * height;
// ½ × b × h
    }
};
int main() {
 ShapeArea shape;
double radius;
cout << "Enter the radius of the circle: ";
 cin >> radius;
cout << "Area of the circle: " << shape.area(radius) << endl;
double length, breadth;
cout << "\nEnter the length and breadth of the rectangle: ";
cin >> length >> breadth;
 cout << "Area of the rectangle: " << shape.area(length, breadth) << endl;
double base, height;
cout << "\nEnter the base and height of the triangle: ";
```

```
cin >> base >> height;     cout << "Area of the triangle: " <<
shape.area(base, height, true) << endl;
return 0;
}
```

Output:



Ques 5. Write a program to demonstrate runtime polymorphism in C++ using a base class Shape and derived classes Circle, Rectangle, and Triangle. The program should use virtual functions to calculate and print the area of each shape based on user input.

Program Code:

```
#include <iostream>
#include <cmath>
using namespace std;
class Shape { public:
virtual void calculateArea() = 0;
virtual ~Shape() {
}
};
```

```
class Circle : public Shape { private:
    double radius;
public:
    Circle(double r) : radius(r) {}
 void calculateArea() override {
double area = M_PI * radius * radius;
cout << "Area of the Circle: " << area << endl;
    }
};
class Rectangle : public Shape {
private:
    double length, breadth;
public:
    Rectangle(double l, double b) : length(l), breadth(b) {}
    void calculateArea() override {
  double area = length * breadth;
cout << "Area of the Rectangle: " << area << endl;
    }
};
class Triangle : public Shape {
private:
    double base, height; public:
    Triangle(double b, double h) : base(b), height(h) {}
void calculateArea()
override {
double area = 0.5 * base * height;
 cout << "Area of the Triangle: " << area << endl;
    }
```

```
}; int main() {
Shape* shape = nullptr;
int choice;
 cout << "Choose a shape to calculate area:\n";
cout << "1. Circle\n2. Rectangle\n3. Triangle\n";
cout << "Enter your choice: ";
 cin >> choice;     switch (choice) {
case 1: {
 double radius;
cout << "Enter the radius of the circle: ";
cin >> radius;
shape = new Circle(radius);
        break;
}
   case 2: {
double length, breadth;
cout << "Enter the length and breadth of the rectangle: ";
cin >> length >> breadth;
 shape = new Rectangle(length, breadth);
        break;
}
case 3: {
double base, height;
cout << "Enter the base and height of the triangle: ";
cin >> base >> height;
shape = new Triangle(base, height);
        break;
}
```

```
default:
cout << "Invalid choice!" << endl;
        return 1;
}
if (shape) {
shape->calculateArea();
delete shape;
return 0;
}
```

Output: