Name Tarun kumar **U ID** 22BCS15293

Section 22BCS_IOT_620-A **Date:** 19 DEC 2024

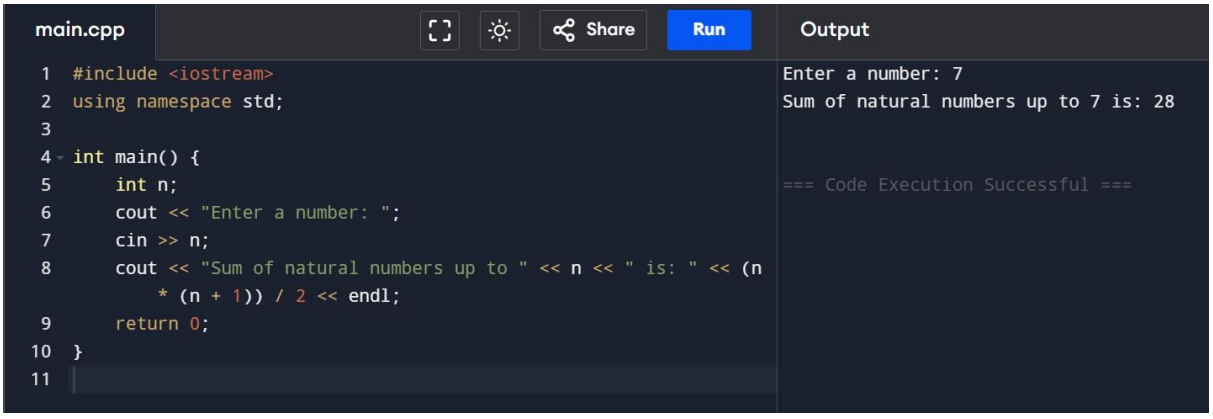
DOMAIN WINTER WINNING CAMP

1) Sum of Natural Numbers up to N

Code:

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

int main() {
   int n;
   cout << "Enter a number: ";
   cin >> n;
   cout << "Sum of natural numbers up to " << n << " is: " << (n * (n + 1)) / 2 << endl;
   return 0;
```

Output:

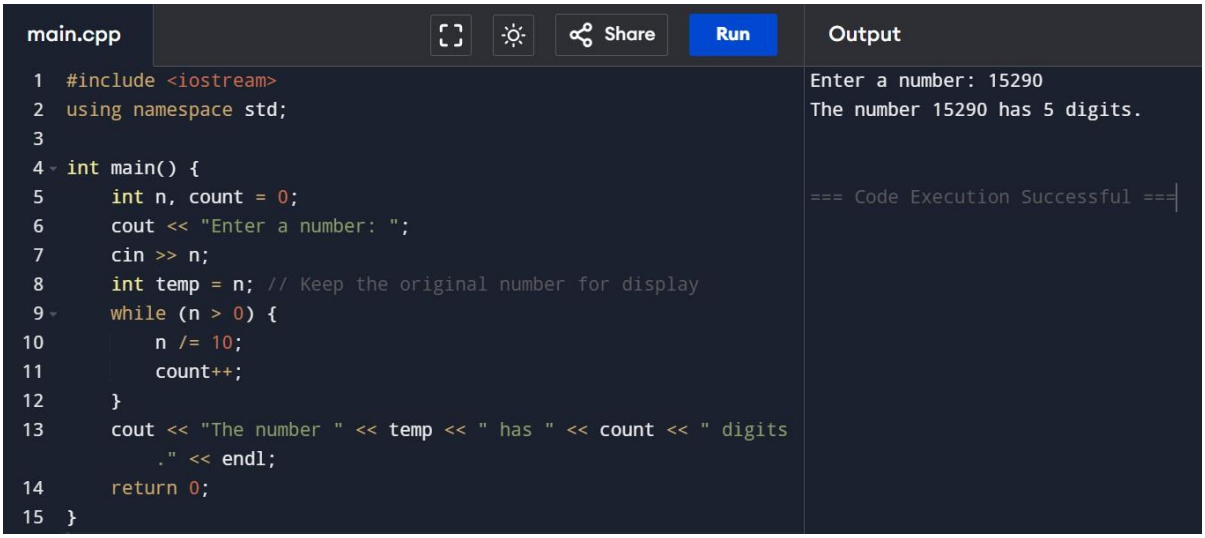


2) Count Digits in a Number

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

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

int main() {
    int n;
    cout << "Enter a positive integer: ";
    cin >> n;

    if (n > 0) {
        int digitCount = countDigits(n);
        cout << "The number of digits in " << n << " is: " << digitCount << endl;
    } else {
```

```
        cout << "Please enter a positive integer!" << endl;
    }
return 0;
}
```

Output:



3) Function overloading

for Area calculation

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

float area(float radius) {
    return 3.14159 * radius * radius;
}

float area(float length, float breadth) {
    return length * breadth;
}

float area(float base, float height, bool triangle) {
```

```
    return 0.5 * base * height;
}

int main() {
    float radius, length, breadth, base, height;

    cout << "Enter the radius of the circle: ";
    cin >> radius;
    cout << "Circle Area: " << area(radius) << endl;

    cout << "Enter the length and breadth of the rectangle: ";
    cin >> length >> breadth;
    cout << "Rectangle Area: " << area(length, breadth) << endl;

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

Output:

main.cpp

Share Run

Output

```
}

int main() {
    float radius, length, breadth, base, height;

    cout << "Enter the radius of the circle: ";
    cin >> radius;
    cout << "Circle Area: " << area(radius) << endl;

    cout << "Enter the length and breadth of the rectangle: ";
    cin >> length >> breadth;
    cout << "Rectangle Area: " << area(length, breadth) << endl
        ;

    cout << "Enter the base and height of the triangle: ";
    cin >> base >> height;
    cout << "Triangle Area: " << area(base, height, true) <<
        endl;

    return 0;
```

```
Enter the radius of the circle: 7
Circle Area: 153.938
Enter the length and breadth of the rectangle: 2 3
Rectangle Area: 6
Enter the base and height of the triangle: 4 5
Triangle Area: 10


=== Code Execution Successful ===
```

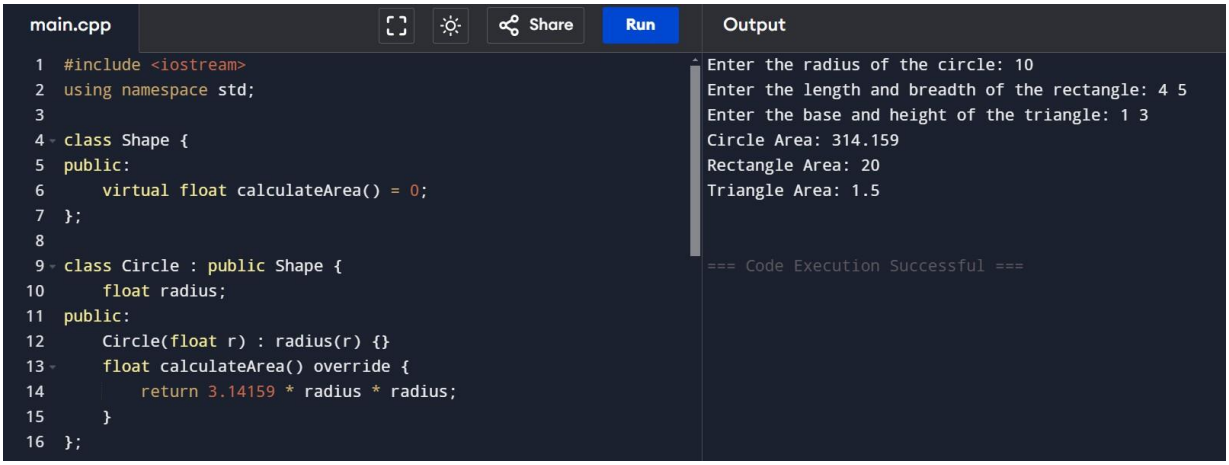
4) Polymorphism with shape Area Calculations .

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

class Shape {
public:
    virtual float calculateArea() = 0;
};

class Circle : public Shape {
    float radius;
public:
    Circle(float r) : radius(r) {}
    float calculateArea() override {
        return 3.14159 * radius * radius;
    }
};

class Rectangle : public Shape {
    float length, breadth;
public:
    Rectangle(float l, float b) : length(l), breadth(b) {}
    float calculateArea() override {
        return length * breadth;
    }
};
```

```
class Triangle : public Shape {
    float base, height;
public:
    Triangle(float b, float h) : base(b), height(h) {}
    float calculateArea() override {
        return 0.5 * base * height;
    }
};

int main() {
    float radius, length, breadth, base, height;

    cout << "Enter the radius of the circle: ";
    cin >> radius;
    Shape* circle = new Circle(radius);

    cout << "Enter the length and breadth of the rectangle: ";
    cin >> length >> breadth;
    Shape* rectangle = new Rectangle(length, breadth);

    cout << "Enter the base and height of the triangle: ";
    cin >> base >> height;
    Shape* triangle = new Triangle(base, height);

    cout << "Circle Area: " << circle->calculateArea() << endl;
    cout << "Rectangle Area: " << rectangle->calculateArea() << endl;
    cout << "Triangle Area: " << triangle->calculateArea() << endl;
```

```
    delete circle;
    delete rectangle;
    delete triangle;

    return 0;
}
```

Output:



5) Multi-level inheritance for vehicle simulation

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

class Vehicle {
protected:
    string brand, model;
    double mileage;
public:
    Vehicle(string b, string m, double mil) : brand(b), model(m), mileage(mil) {}
    virtual void displayDetails() {
        cout << "Brand: " << brand << ", Model: " << model << ", Mileage: " <<
mileage << " miles" << endl;
    }
```

```
};

class Car : public Vehicle {
protected:
    double fuel, distance;
public:
    Car(string b, string m, double mil, double f, double d) : Vehicle(b, m, mil),
fuel(f), distance(d) {}
    double calculateFuelEfficiency() {
        return distance / fuel;
    }
    void displayDetails() override {
        Vehicle::displayDetails();
        cout << "Fuel Efficiency: " << calculateFuelEfficiency() << " miles/gallon" <<
endl;
    }
};

class ElectricCar : public Car {
    double batteryCapacity, efficiency;
public:
    ElectricCar(string b, string m, double mil, double bc, double eff)
        : Car(b, m, mil, 0, 0), batteryCapacity(bc), efficiency(eff) {}
    double calculateRange() {
        return batteryCapacity * efficiency;
    }
    void displayDetails() override {
        Vehicle::displayDetails();
        cout << "Range: " << calculateRange() << " miles" << endl;
```

```
    }
};

int main() {
    string brand, model;
    double mileage, fuel, distance, batteryCapacity, efficiency;

    cout << "Enter Car details:\nBrand: ";
    cin >> brand;
    cout << "Model: ";
    cin >> model;
    cout << "Mileage (miles): ";
    cin >> mileage;
    cout << "Fuel (gallons): ";
    cin >> fuel;
    cout << "Distance Covered (miles): ";
    cin >> distance;

    Car car(brand, model, mileage, fuel, distance);
    car.displayDetails();

    cout << "\nEnter Electric Car details:\nBrand: ";
    cin >> brand;
    cout << "Model: ";
    cin >> model;
    cout << "Mileage (miles): ";
    cin >> mileage;
    cout << "Battery Capacity (kWh): ";
    cin >> batteryCapacity;
```

```
    cout << "Efficiency (miles/kWh): ";
    cin >> efficiency;

    ElectricCar eCar(brand, model, mileage, batteryCapacity, efficiency);
    eCar.displayDetails();

    return 0;
}
```

Output: