

**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:

main.cpp	Output
<pre>1 #include &lt;iostream&gt; 2 using namespace std; 3 4 int main() { 5     int n; 6     cout &lt;&lt; "Enter a number: "; 7     cin &gt;&gt; n; 8     cout &lt;&lt; "Sum of natural numbers up to " &lt;&lt; n &lt;&lt; " is: " &lt;&lt; (n         * (n + 1)) / 2 &lt;&lt; endl; 9     return 0; 10 } 11</pre>	<pre>Enter a number: 7 Sum of natural numbers up to 7 is: 28  === Code Execution Successful ===</pre>

## 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:

main.cpp	Output
<pre> 1  #include &lt;iostream&gt; 2  using namespace std; 3 4  int main() { 5      int n, count = 0; 6      cout &lt;&lt; "Enter a number: "; 7      cin &gt;&gt; n; 8      int temp = n; // Keep the original number for display 9      while (n &gt; 0) { 10         n /= 10; 11         count++; 12     } 13     cout &lt;&lt; "The number " &lt;&lt; temp &lt;&lt; " has " &lt;&lt; count &lt;&lt; " digits 14         ." &lt;&lt; endl; 15     return 0; 16 } </pre>	<pre> Enter a number: 15290 The number 15290 has 5 digits.  === Code Execution Successful === </pre>

## 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	Output
<pre>14 } 15 16 int main() { 17     float radius, length, breadth, base, height; 18 19     cout &lt;&lt; "Enter the radius of the circle: "; 20     cin &gt;&gt; radius; 21     cout &lt;&lt; "Circle Area: " &lt;&lt; area(radius) &lt;&lt; endl; 22 23     cout &lt;&lt; "Enter the length and breadth of the rectangle: "; 24     cin &gt;&gt; length &gt;&gt; breadth; 25     cout &lt;&lt; "Rectangle Area: " &lt;&lt; area(length, breadth) &lt;&lt; endl 26         ; 27 28     cout &lt;&lt; "Enter the base and height of the triangle: "; 29     cin &gt;&gt; base &gt;&gt; height; 30     cout &lt;&lt; "Triangle Area: " &lt;&lt; area(base, height, true) &lt;&lt; 31         endl; 32 33     return 0; }</pre>	<pre>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 ===</pre>

#### **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:

The screenshot shows a C++ IDE with a file named 'main.cpp'. The code defines a base class 'Shape' with a virtual function 'calculateArea()' and two derived classes: 'Circle' and 'Rectangle'. The 'Circle' class has a 'radius' attribute and a constructor. The 'Rectangle' class has 'length' and 'breadth' attributes. The program prompts the user to enter values for these attributes and then calculates and displays the areas. The output shows the calculated areas for a circle with radius 10, a rectangle with length 4 and breadth 5, and a triangle with base 1 and height 3.

```

main.cpp
1 #include <iostream>
2 using namespace std;
3
4 class Shape {
5 public:
6     virtual float calculateArea() = 0;
7 };
8
9 class Circle : public Shape {
10     float radius;
11 public:
12     Circle(float r) : radius(r) {}
13     float calculateArea() override {
14         return 3.14159 * radius * radius;
15     }
16 };

```

Output:

```

Enter the radius of the circle: 10
Enter the length and breadth of the rectangle: 4 5
Enter the base and height of the triangle: 1 3
Circle Area: 314.159
Rectangle Area: 20
Triangle Area: 1.5

=== Code Execution Successful ===

```

## 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:

main.cpp	Output
<pre>1 #include &lt;iostream&gt; 2 using namespace std; 3 4 class Vehicle { 5 protected: 6     string brand, model; 7     double mileage; 8 public: 9     Vehicle(string b, string m, double mil) : brand(b), model(m         ), mileage(mil) {} 10    virtual void displayDetails() { 11        cout &lt;&lt; "Brand: " &lt;&lt; brand &lt;&lt; ", Model: " &lt;&lt; model &lt;&lt; "             , Mileage: " &lt;&lt; mileage &lt;&lt; " miles" &lt;&lt; endl; 12    } 13 }; 14 15 class Car : public Vehicle {</pre>	<pre>Enter Car details: Brand: BMW Model: 2024 Mileage (miles): 20 Fuel (gallons): 40 Distance Covered (miles): 35 Brand: BMW, Model: 2024, Mileage: 20 miles Fuel Efficiency: 0.875 miles/gallon  Enter Electric Car details: Brand: mg Model: 2024 Mileage (miles): 15 Battery Capacity (kWh): 50.3 Efficiency (miles/kWh): 461 Brand: mg, Model: 2024, Mileage: 15 miles Range: 23188.3 miles</pre>