Cal simple

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

Class Calc {
Public:
  Void add(int a, int b) { cout << a + b; }
};

Int main() {
  Calc c;
  c.add(5, 3);  // Example: 5 + 3
  return 0;
}
```

Method overload

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

class Calc {
public:
  void add(int a, int b) {
    cout << "Sum (int): " << a + b << endl;
  }

  void add(float a, float b) {
```

```
        cout << "Sum (float): " << a + b << endl;
    }
};

int main() {
    Calc c;
    c.add(5, 3);       // Calls int version
    c.add(2.5f, 1.5f);  // Calls float version
    return 0;
}
```

Call by value

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

class Calc {
public:
    void add(int a, int b) {
        a = a + 10; // Changes here won't affect original values
        b = b + 5;
        cout << "Sum inside function: " << a + b << endl;
    }
};

int main() {
    int x = 5, y = 3;
```

```
    Calc c;
    c.add(x, y);
    cout << "Original values: " << x << " " << y << endl;
    return 0;
}
```

Simple

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

class Parent {
public:
    void show() {
        cout << "This is Parent class\n";
    }
};

class Child : public Parent {
};

int main() {
    Child c;
    c.show();
    return 0;
}
```

Multiple

```
#include <iostream>
```

```
Using namespace std;

Class A {
Public:
   Void fromA() {
      Cout << “From class A\n”;
   }
};

Class B {
Public:
   Void fromB() {
      Cout << “From class B\n”;
   }
};

Class C : public A, public B {
};

Int main() {
   C obj;
   Obj.fromA();
   Obj.fromB();
   Return 0;
}
```

Multilevel

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

class Grandparent {
public:
   void greet() {
      cout << "Hello from Grandparent\n";
   }
};

class Parent : public Grandparent {
};

class Child : public Parent {
};

int main() {
   Child c;
   c.greet();
   return 0;
}
```

Program: Class and Object (Student Info)

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

Class Student {
```

```
Public:
   Int rollNo;
   Int age;

   Void input() {
      Cout << “Enter Roll Number: “;
      Cin >> rollNo;
      Cout << “Enter Age: “;
      Cin >> age;
   }

   Void display() {
      Cout << “Roll Number: “ << rollNo << endl;
      Cout << “Age: “ << age << endl;
   }
};

Int main() {
   Student s1;  // Object of class Student
   S1.input();  // Call input function
   S1.display(); // Call display function

   Return 0;
}
```

Bank

```
#include <iostream>
```

```
Using namespace std;
Class BankAccount {
Public:
   String name;
   Int accNo;
   Float balance;

   Void input() {
      Cout << “Enter Account Holder Name: “;
      Cin >> name;
      Cout << “Enter Account Number: “;
      Cin >> accNo;
      Cout << “Enter Initial Balance: “;
      Cin >> balance;
   }

   Void display() {
      Cout << “\n--- Account Details ---” << endl;
      Cout << “Name: “ << name << endl;
      Cout << “Account Number: “ << accNo << endl;
      Cout << “Balance: ₹” << balance << endl;
   }
};

Int main() {
   BankAccount b1;
```

```
    B1.input();
    B1.display();
    Return 0;
}
```

Odd even

```
#include <iostream>
Using namespace std;
Int main() {
    Int num;
    Cout << “Enter a number: “;
    Cin >> num;

    // Check Positive or Negative
    If (num > 0)
        Cout << “The number is Positive” << endl;
    Else if (num < 0)
        Cout << “The number is Negative” << endl;
    Else
        Cout << “The number is Zero” << endl;

    // Check Even or Odd
    If (num % 2 == 0)
        Cout << “The number is Even” << endl;
    Else
        Cout << “The number is Odd” << endl;
```

```
    Return 0;
}
```

Multi table

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

Int main() {
    Int num;
    Cout << “Enter a number: “;
    Cin >> num;

    Cout << “Multiplication Table of “ << num << “:\n”;

    For (int I = 1; I <= 10; i++) {
        Cout << num << “ x “ << I << “ = “ << num * I << endl;
    }

    Return 0;
}
```

Square area cone

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

Int main() {
```

```
    Float side = 4;
    Float length = 5, breadth = 3;
    Float radius = 2, slantHeight = 6;
    Float squareArea, rectangleArea, coneArea;

    squareArea = side * side;
    rectangleArea = length * breadth;
    coneArea = 3.14 * radius * (radius + slantHeight);  // πr(r + l)

    cout << “Area of Square: “ << squareArea << endl;
    cout << “Area of Rectangle: “ << rectangleArea << endl;
    cout << “Surface Area of Cone: “ << coneArea << endl;

    return 0;
}
```

Strong, array, arithmetic

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

int main() {
    try {
        int a = 10, b = 0;
        if (b == 0) throw "Divide by zero!";
        cout << a / b << endl;
```

```
    } catch (const char* e) {
        cout << e << endl;
    }

    try {
        int arr[3] = {1,2,3};
        if (3 >= 3) throw "Array out of bounds!";
        cout << arr[3] << endl;
    } catch (const char* e) {
        cout << e << endl;
    }

    try {
        string s = "";
        if (s == "") throw "Empty string!";
        cout << s << endl;
    } catch (const char* e) {
        cout << e << endl;
    }

    return 0;
}
```