Program-1

Write a program to calculate the sum of the digits of a number.

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

int main()
{
   int number, remainder, sum = 0;
   cout << "\n Enter a number:";
   cin >> number;
   while (number != 0)
   {
      remainder = number % 10;
      sum = sum + remainder;
      number = number / 10;
   }
   cout << "\n Sum of digits is " << sum;
   return 0;
}
```

OUTPUT:-

```
Enter a number: 5402

Sum of digits is 11
```

Program-2

Write a program to reverse a number.

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

int main()
{
   int n, Rem, Reverse = 0;
   cout << "\n Enter a number :";
   cin >> n;
   while (n != 0)
   {
      Rem = n % 10;
      Reverse = Reverse * 10 + Rem;
      n = n / 10;
   }
   cout << " Reversed Number is: " << Reverse << endl;
   return 0;
}
```

OUTPUT:-

```
Enter a number :5402
Reversed Number is: 2045
```

Program-3

Write a program to create multiplication table of a number.

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

int main()
{
   int num;
   cout << "Enter number to find multiplication table of:";
   cin >> num;
   for (int a = 1; a <= 10; a++)
   {
      cout << num << " *  " << a << "  =  " << num * a << endl;
   }
   return 0;
}
```

OUTPUT:-

```
Enter number to find multiplication table of: 5
5 *  1  =  5
5 *  2  =  10
5 *  3  =  15
5 *  4  =  20
5 *  5  =  25
5 *  6  =  30
5 *  7  =  35
5 *  8  =  40
5 *  9  =  45
5 *  10  =  50
```

Program-4

Write a program to print first n prime numbers.

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

void prime(int n)
{
   int factors;
   cout << "Prime numbers are...";
   for (int i = 1; i <= n; i++)
   {
      factors = 0;
      for (int j = 1; j <= i; j++)
      {
         if (i % j == 0)
            factors = factors + 1;
      }
      if (factors <= 2)
         cout << i << "\t";
   }
}

int main()
{
   int n;
   cout << "\nEnter a integer value :";
   cin >> n;
   prime(n);

   return 0;
}
```

OUTPUT:-

```
Enter a integer value :40
Prime numbers are...1   2       3       5       7       11      13      17      19      23      29      31      37
```

Program-5

Write a program to implement basic mathematical operations using class and objects.

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

class MathOperation {
   private:
      float num1, num2;
   public:
      void input() {
         cout << "\nEnter two numbers: ";
         cin >> num1 >> num2;
      }
      float add() { return num1 + num2; }
      float subtract() { return num1 - num2; }
      float multiply() { return num1 * num2; }
      float divide() { return (num2 != 0) ? num1 / num2 : 0; }
};

int main() {
   MathOperation mo;
   mo.input();
   cout << "Addition: " << mo.add() << endl;
   cout << "Subtraction: " << mo.subtract() << endl;
   cout << "Multiplication: " << mo.multiply() << endl;
   cout << "Division: " << mo.divide() << endl;
   return 0;
}
```

OUTPUT:-

```
Enter two numbers: 4 5
Addition: 9
Subtraction: -1
Multiplication: 20
Division: 0.8
```

Program-6

Write a program to show use of loops using class and member functions.

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

class LoopDemo {
   private:
      int n;
   public:
      void input() {
         cout << "\nEnter the value of n: ";
         cin >> n;
      }
      void printNumbers() {
         for (int i = 1; i <= n; i++) {
            cout << i << " ";
         }
         cout << endl;
      }
};

int main() {
   LoopDemo loopObj;
   loopObj.input();
   cout<<"After looping....."<<endl;
   loopObj.printNumbers();
   return 0;
}
```

OUTPUT:-

```
Enter the value of n: 6
After looping.....
1 2 3 4 5 6
```

Program-7

Write a program to implement the array of objects.

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

class Student {
   private:
      string name;
      int rollNo;
   public:
      void input() {
         cout << "Enter name and roll number: ";
         cin >> name >> rollNo;
      }
      void display() {
         //cout << name <<"           "<< rollNo<<endl;
         cout << setw(12) << left << name << setw(10) << right << rollNo<< endl;
      }
};

int main() {
   int n;
   cout << "Enter the number of students: ";
   cin >> n;

   Student students[n];
   for(int i = 0; i < n; i++) {
      cout << "Enter details of student " << i+1 << ":" << endl;
      students[i].input();
   }

   cout << "\nDetails of students:" << endl;
   cout<<"Name           Roll No"<<endl;
   for(int i = 0; i < n; i++) {
      students[i].display();
   }
   return 0;
}
```

OUTPUT:-

```
Enter the number of students: 5
Enter details of student 1:
Enter name and roll number: Vaibhav 5401
Enter details of student 2:
Enter name and roll number: Vani 5402
Enter details of student 3:
Enter name and roll number: Shweta 5403
Enter details of student 4:
Enter name and roll number: Rahul 5404
Enter details of student 5:
Enter name and roll number: Sonam 5405

Details of students:
Name            Roll No
Vaibhav           5401
Vani              5402
Shweta            5403
Rahul             5404
Sonam             5405
```

Program-8

Write a program to pass objects as arguments to the member function of class.

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

class Complex {
   public:
      float real, imag;
      void input() {
         cout << "Enter real and imaginary parts: ";
         cin >> real >> imag;
      }
      void addComplex(Complex c1, Complex c2) {
         real = c1.real + c2.real;
         imag = c1.imag + c2.imag;
      }
      void display() {
         cout <<real << " + " << imag << "i" << endl;
      }
};

int main() {
   Complex c1, c2, sum;
   cout << "Enter first complex number:" << endl;
   c1.input();
   cout << "Enter second complex number:" << endl;
   c2.input();

   sum.addComplex(c1, c2);
   cout<<"Sum of "<<c1.real<<" + "<<c1.imag<<"i"<<" and "<<c2.real<<" + "<<c2.imag<<"i"<<" is ";
   sum.display();

   return 0;
}
```

OUTPUT:-

```
Enter first complex number:
Enter real and imaginary parts: 3 4
Enter second complex number:
Enter real and imaginary parts: 2 5
Sum of 3 + 4i and 2 + 5i is 5 + 9i
```

Program-9

Write a program to implement function overloading using class and objects.

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

class Calculator {
   public:
      int add(int a, int b) { return a + b; }
      int add(int a, int b, int c) { return a + b + c; }
      float add(float a, float b) { return a + b; }
};

int main() {
   Calculator calc;
   cout << "Addition of two integers(using add(int a, int b)): " << calc.add(5, 10) <<endl;
   cout << "Addition of three integers(using int(int a, int b, int c)): " << calc.add(5, 10, 15) << endl;
   cout << "Addition of two floats(using int(float a, float b)): " << calc.add(5.5f, 10.5f) << endl;
   return 0;
}
```

OUTPUT:-

```
Addition of two integers(using add(int a, int b)): 15
Addition of three integers(using int(int a, int b, int c)): 30
Addition of two floats(using int(float a, float b)): 16
```

Program-10

Write a program to understand the concept of Constructors, Copy Constructors, Default Constructors and Destructors.

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

class Sample {
private:
   int value;
public:
   // Default constructor
   Sample() {
      Value = 0;
      cout << "Default constructor called, value = " << value << endl;
   }

   // Parameterized constructor
   Sample(int v) {
      Value = v;
      cout << "Parameterized constructor called with value = " << value << endl;
   }

   // Copy constructor
   Sample(const Sample &obj) {
      Value = obj.value;
      cout << "Copy constructor called, copied value = " << value << endl;
   }

   // Destructor
   ~Sample() {
      cout << "Destructor called for value = " << value << endl;
   }

   // Display function
   void display() const {
      cout << "Value: " << value << endl;
   }
};

int main() {
   Sample s1;      // Default constructor
   Sample s2(20);  // Parameterized constructor
   Sample s3(s2);  // Copy constructor

   s1.display();
   s2.display();
```

```
    s3.display();

    return 0;
}
```

OUTPUT:-

```
Default constructor called, value = 0
Parameterized constructor called with value = 20
Copy constructor called, copied value = 20
Value: 0
Value: 20
Value: 20
Destructor called for value = 20
Destructor called for value = 20
Destructor called for value = 0
```

Program-11

Write a program to implement overloading of unary minus operator using member function.

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

class Number {
   private:
      int value;
   public:
      Number(int v){
         value = v;
      }
      void operator-(){
         value = -value;
      }
      void display() {
         cout << "Value: " << value << endl;
      }
};

int main() {
   Number num(50);
   cout << "Original value: ";
   num.display();

   -num;
   cout << "After applying unary minus: ";
   num.display();

   return 0;
}
```

OUTPUT:-

```
Original value: Value: 50
After applying unary minus: Value: -50
```

Program-12

Write a program to implement overloading of binary operators using friend function.

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

class Complex {
   private:
      float real, imag;
   public:
      Complex(float r = 0, float i = 0) : real(r), imag(i) {}
      friend Complex operator+(Complex c1, Complex c2);
      void display() { cout << real << " + " << imag << "i" << endl; }
};

Complex operator+(Complex c1, Complex c2) {
   Complex temp;
   temp.real = c1.real + c2.real;
   temp.imag = c1.imag + c2.imag;
   return temp;
}

int main() {
   Complex c1(3.5, 2.5), c2(1.5, 4.5), sum;

   sum = c1 + c2;

   cout << "First complex number: ";
   c1.display();
   cout << "Second complex number: ";
   c2.display();
   cout << "Sum of complex numbers: ";
   sum.display();

   return 0;
}
```

OUTPUT:-

```
First complex number: 3.5 + 2.5i
Second complex number: 1.5 + 4.5i
Sum of complex numbers: 5 + 7i
```