

MODULE: 4.2 (Programing with C++)

1. WAP to create simple calculator using class.

Code:

```
#include <iostream>
#include <math.h>
using namespace std;

class Calculator
{
float a, b;
public:

    void result()
    {
        cout << "Enter First Number: ";
        cin >> a;
        cout << "Enter Second Number: ";
        cin >> b;
    }

    float add()
    {
        return a + b;
    }
    float sub()
    {
        return a - b;
    }
    float mul()
    {
        return a * b;
    }
    float div()
    {
        if (b == 0)
        {
            cout << "Division By Zero" <<
```

```

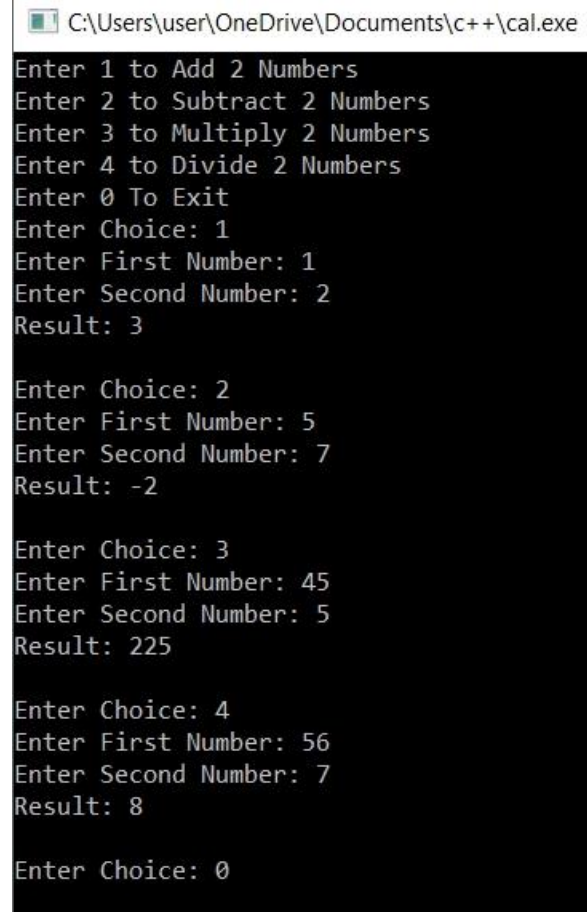
                                endl;
                                return INFINITY;
                            }
                        else
                        {
                            return a / b;
                        }
                    }
                };
int main()
{
    int ch;
    Calculator c;
    cout << "Enter 1 to Add 2 Numbers" <<
        "\nEnter 2 to Subtract 2 Numbers" <<
        "\nEnter 3 to Multiply 2 Numbers" <<
        "\nEnter 4 to Divide 2 Numbers" <<
        "\nEnter 0 To Exit";

    do
    {
        cout << "\nEnter Choice: ";
        cin >> ch;
        switch (ch)
        {
            case 1:
                c.result();
                cout << "Result: " << c.add() << endl;
                break;
            case 2:
                c.result();
                cout << "Result: " << c.sub() << endl;
                break;
            case 3:
                c.result();
                cout << "Result: " << c.mul() << endl;
                break;
            case 4:
                c.result();
                cout << "Result: " << c.div() << endl;
                break;
        }
    }

```

```
    } while (ch >= 1 && ch <= 4);  
  
    return 0;  
}
```

Output:



```
C:\Users\user\OneDrive\Documents\c++\cal.exe  
Enter 1 to Add 2 Numbers  
Enter 2 to Subtract 2 Numbers  
Enter 3 to Multiply 2 Numbers  
Enter 4 to Divide 2 Numbers  
Enter 0 To Exit  
Enter Choice: 1  
Enter First Number: 1  
Enter Second Number: 2  
Result: 3  
  
Enter Choice: 2  
Enter First Number: 5  
Enter Second Number: 7  
Result: -2  
  
Enter Choice: 3  
Enter First Number: 45  
Enter Second Number: 5  
Result: 225  
  
Enter Choice: 4  
Enter First Number: 56  
Enter Second Number: 7  
Result: 8  
  
Enter Choice: 0
```

2. Define a class to represent a bank account. Include the following members:

1. Data Member:

- Name of the depositor**
- Account Number**
- Type of Account**
- Balance amount in the account**

2. Member Functions

- To assign values**
- To deposited an amount**
- To withdraw an amount after checking balance**
- To display name and balance**

Code:

```
#include <iostream>
#include <string>
```

```
class BankAccount {
private:
```

```
    std::string depositorName;
    std::string accountNumber;
    std::string accountType;
    double balance;
```

```
public:
```

```
    BankAccount(const std::string& name, const std::string& accNumber, const
std::string& accType) {
        depositorName = name;
        accountNumber = accNumber;
        accountType = accType;
```

```

        balance = 0.0;
    }
    void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            std::cout << "Deposited $" << amount << ". New balance: $" << balance
<< std::endl;
        } else {
            std::cout << "Invalid deposit amount. Amount must be greater than
zero." << std::endl;
        }
    }
    void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            std::cout << "Withdrew $" << amount << ". New balance: $" << balance
<< std::endl;
        } else if (amount <= 0) {
            std::cout << "Invalid withdrawal amount. Amount must be greater than
zero." << std::endl;
        } else {
            std::cout << "Insufficient balance for withdrawal." << std::endl;
        }
    }

    void display() {
        std::cout << "Account Holder: " << depositorName << std::endl;
        std::cout << "Account Number: " << accountNumber << std::endl;
        std::cout << "Account Type: " << accountType << std::endl;
        std::cout << "Balance: $" << balance << std::endl;
    }
};

int main() {
    BankAccount account("John Doe", "12345", "Savings");

    account.display();

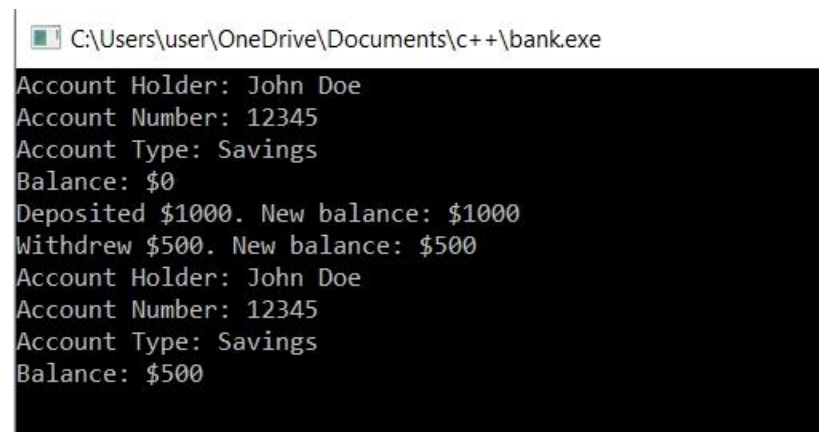
    account.deposit(1000);

    account.withdraw(500);

```

```
account.display();  
  
return 0;  
}
```

Output:



The screenshot shows a Windows command prompt window with the title bar "C:\Users\user\OneDrive\Documents\c++\bank.exe". The output of the program is displayed in white text on a black background. The output shows the account holder's name, account number, and account type, followed by the initial balance of \$0. Then, a deposit of \$1000 is made, resulting in a new balance of \$1000. Next, a withdrawal of \$500 is made, resulting in a new balance of \$500. Finally, the account details are displayed again, showing the updated balance of \$500.

```
C:\Users\user\OneDrive\Documents\c++\bank.exe  
Account Holder: John Doe  
Account Number: 12345  
Account Type: Savings  
Balance: $0  
Deposited $1000. New balance: $1000  
Withdrew $500. New balance: $500  
Account Holder: John Doe  
Account Number: 12345  
Account Type: Savings  
Balance: $500
```

3. Write a program to find the multiplication values and the cubic values using inline function

Code:

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

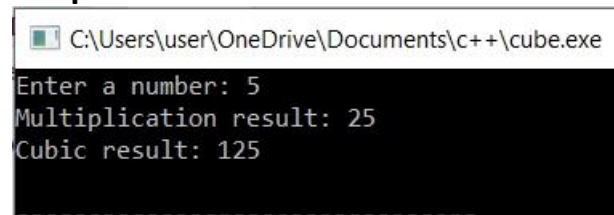
inline double multiply(double a, double b) {
    return a * b;
}
inline double cube(double x) {
    return x * x * x;
}

int main() {
    double num;
    cout << "Enter a number: ";
    cin >> num;
    double multiplicationResult = multiply(num, 5.0);
    cout << "Multiplication result: " << multiplicationResult << std::endl;

    double cubicResult = cube(num);
    cout << "Cubic result: " << cubicResult << std::endl;

    return 0;
}
```

Output:



```
C:\Users\user\OneDrive\Documents\c++\cube.exe
Enter a number: 5
Multiplication result: 25
Cubic result: 125
-----
```

4. Write a program of Addition, Subtraction, Division, Multiplication using constructor.

Code:

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

class Calculator {
private:
    double num1;
    double num2;

public:
    Calculator(double a, double b) {
        num1 = a;
        num2 = b;
    }
    double add() {
        return num1 + num2;
    }
    double subtract() {
        return num1 - num2;
    }
    double divide() {
        if (num2 != 0) {
            return num1 / num2;
        } else {
            std::cout << "Error: Division by zero is not allowed." << std::endl;
            return 0;
        }
    }
    double multiply() {
        return num1 * num2;
    }
};

int main() {
    double a, b;
    cout << "Enter the first number: ";
```

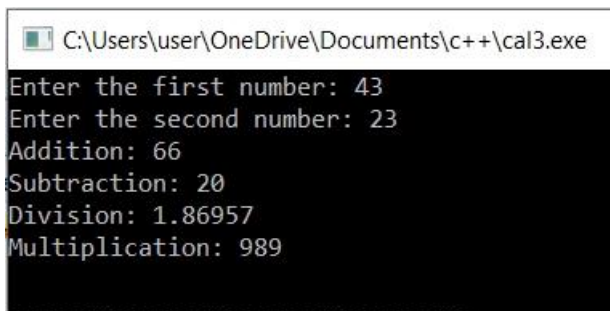


```
cin >> a;
cout << "Enter the second number: ";
cin >> b;

Calculator calculator(a, b);
cout << "Addition: " << calculator.add() << endl;
cout << "Subtraction: " << calculator.subtract() << endl;
cout << "Division: " << calculator.divide() << endl;
cout << "Multiplication: " << calculator.multiply() << endl;

return 0;
}
```

Output:



```
C:\Users\user\OneDrive\Documents\c++\cal3.exe
Enter the first number: 43
Enter the second number: 23
Addition: 66
Subtraction: 20
Division: 1.86957
Multiplication: 989
```

5. Assume a class cricketer is declared. Declare a derived class batsman from cricketer. Data member of batsman. Total runs, Average runs and best performance. Member functions input data, calculate average runs, Display data. (Single Inheritance)

Code:

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

class Cricketer {
protected:
    string name;
    int age;

public:
    Cricketer(const string& playerName, int playerAge) {
        name = playerName;
        age = playerAge;
    }
    void inputData() {
        cout << "Enter the name of the cricketer: ";
        cin >> name;
        cout << "Enter the age of the cricketer: ";
        cin >> age;
    }
    void displayData() {
        cout << "Name: " << name << endl;
        cout << "Age: " << age << " years" << endl;
    }
};

class Batsman : public Cricketer {
private:
    int totalRuns;
    double averageRuns;
    int bestPerformance;
```

```

public:
    Batsman(const std::string& playerName, int playerAge, int runs, double avg,
int best) : Cricketer(playerName, playerAge) {
        totalRuns = runs;
        averageRuns = avg;
        bestPerformance = best;
    }
    void calculateAverageRuns() {
        averageRuns = static_cast<double>(totalRuns) / 100;
    }
    void displayData() {
        Cricketer::displayData();
        cout << "Total Runs: " << totalRuns << endl;
        cout << "Average Runs: " << averageRuns << endl;
        cout << "Best Performance: " << bestPerformance << " runs in an inning"
<< endl;
    }
};

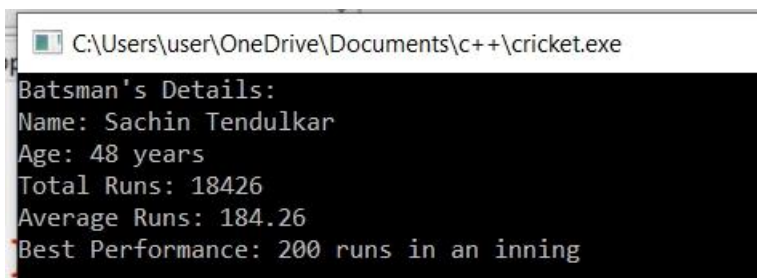
int main() {
    Batsman batsman("Sachin Tendulkar", 48, 18426, 0.0, 200);
    batsman.calculateAverageRuns();

    cout << "Batsman's Details:" << endl;
    batsman.displayData();

    return 0;
}

```

Output:



```

C:\Users\user\OneDrive\Documents\c++\cricket.exe
Batsman's Details:
Name: Sachin Tendulkar
Age: 48 years
Total Runs: 18426
Average Runs: 184.26
Best Performance: 200 runs in an inning

```

6. Assume that the test results of a batch of students are stored in three different classes. Class Students are storing the roll number. Class Test stores the marks obtained in two subjects and class result contains the total marks obtained in the test. The class result can inherit the details of the marks obtained in the test and roll number of students. (Multilevel Inheritance)

Code:

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

class Students {
protected:
    int rollNumber;

public:
    Students(int roll) : rollNumber(roll) {}
    void inputRollNumber() {
        cout << "Enter the roll number: ";
        cin >> rollNumber;
    }
    void displayRollNumber() {
        cout << "Roll Number: " << rollNumber << endl;
    }
};

class Test : public Students {
protected:
    int subject1Marks;
    int subject2Marks;

public:
```

```

    Test(int roll, int marks1, int marks2) : Students(roll), subject1Marks(marks1),
subject2Marks(marks2) {}
    void inputMarks() {
        inputRollNumber();
        cout << "Enter marks in Subject 1: ";
        cin >> subject1Marks;
        cout << "Enter marks in Subject 2: ";
        cin >> subject2Marks;
    }
    void displayMarks() {
        displayRollNumber();
        cout << "Marks in Subject 1: " << subject1Marks << endl;
        cout << "Marks in Subject 2: " << subject2Marks << endl;
    }
};

class Result : public Test {
public:
    Result(int roll, int marks1, int marks2) : Test(roll, marks1, marks2) {}
    void calculateAndDisplayTotalMarks() {
        int totalMarks = subject1Marks + subject2Marks;
        displayMarks();
        std::cout << "Total Marks: " << totalMarks << endl;
    }
};

int main() {
    int roll, marks1, marks2;
    cout << "Enter student details:" << endl;
    cout << "-----" << endl;
    cout << "Enter roll number: ";Output
    cin >> roll;
    cout << "Enter marks in Subject 1: ";
    cin >> marks1;
    cout << "Enter marks in Subject 2: ";
    cin >> marks2;

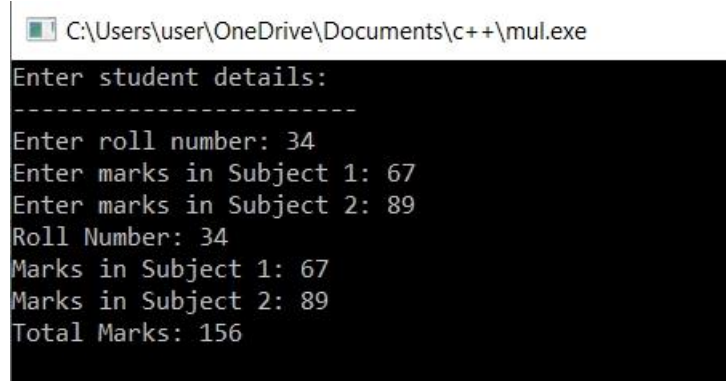
    Result result(roll, marks1, marks2);
    result.calculateAndDisplayTotalMarks();

    return 0;
}

```

```
}
```

Output:



```
C:\Users\user\OneDrive\Documents\c++\mul.exe
Enter student details:
-----
Enter roll number: 34
Enter marks in Subject 1: 67
Enter marks in Subject 2: 89
Roll Number: 34
Marks in Subject 1: 67
Marks in Subject 2: 89
Total Marks: 156
```

7. Write a program to Mathematic operation like Addition, Subtraction, Multiplication, Division Of two number using different parameters and Function Overloading.

Code:

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

class MathOperations {
public:
    int add(int a, int b) {
        return a + b;
    }
    double add(double a, double b) {
        return a + b;
    }
    int subtract(int a, int b) {
        return a - b;
    }
    double subtract(double a, double b) {
        return a - b;
    }
    int multiply(int a, int b) {
        return a * b;
    }
    double multiply(double a, double b) {
        return a * b;
    }
    int divide(int a, int b) {
        if (b != 0) {
            return a / b;
        } else {
            cout << "Error: Division by zero." << endl;
            return 0;
        }
    }
    double divide(double a, double b) {
```

```

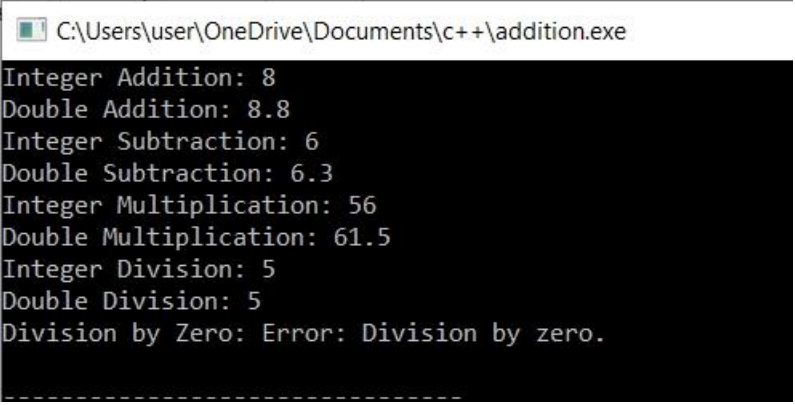
        if (b != 0.0) {
            return a / b;
        } else {
            cout << "Error: Division by zero." << endl;
            return 0.0;
        }
    }
};

int main() {
    MathOperations math;
    cout << "Integer Addition: " << math.add(5, 3) << endl;
    cout << "Double Addition: " << math.add(5.5, 3.3) << endl;
    cout << "Integer Subtraction: " << math.subtract(10, 4) << endl;
    cout << "Double Subtraction: " << math.subtract(10.5, 4.2) << endl;
    cout << "Integer Multiplication: " << math.multiply(7, 8) << endl;
    cout << "Double Multiplication: " << math.multiply(7.5, 8.2) << endl;
    cout << "Integer Division: " << math.divide(20, 4) << endl;
    cout << "Double Division: " << math.divide(20.0, 4.0) << endl;
    cout << "Division by Zero: ";math.divide(10, 0);

    return 0;
}

```

Output:



```

C:\Users\user\OneDrive\Documents\c++\addition.exe
Integer Addition: 8
Double Addition: 8.8
Integer Subtraction: 6
Double Subtraction: 6.3
Integer Multiplication: 56
Double Multiplication: 61.5
Integer Division: 5
Double Division: 5
Division by Zero: Error: Division by zero.

```


8. Write a Program of Two 1D Matrix Addition using Operator Overloading

Code:

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

class Matrix
{
    int a[3][3];
public:
    void accept();
    void display();
    void operator +(Matrix x);
};

void Matrix::accept()
{
    cout<<"\n Enter Matrix Element (3 X 3) : \n";
    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            cout<<" ";
            cin>>a[i][j];
        }
    }
}

void Matrix::display()
{
    for(int i=0; i<3; i++)
    {
        cout<<" ";
        for(int j=0; j<3; j++)
        {
            cout<<a[i][j]<<"\t";
        }
        cout<<"\n";
    }
}

void Matrix::operator +(Matrix x)
```

```

{
    int mat[3][3];
    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            mat[i][j]=a[i][j]+x.a[i][j];
        }
    }
    cout<<"\n Addition of Matrix : \n\n";
    for(int i=0; i<3; i++)
    {
        cout<<" ";
        for(int j=0; j<3; j++)
        {
            cout<<mat[i][j]<<"\t";
        }
        cout<<"\n";
    }
}

int main()
{
    Matrix m,n;
    m.accept();
    n.accept();
    cout<<"\n First Matrix : \n\n";
    m.display();
    cout<<"\n Second Matrix : \n\n";
    n.display();
    m+n;
    return 0;
}

```

Output:

```
C:\Users\user\OneDrive\Documents\c++\matrix.exe

Enter Matrix Element (3 X 3) :
1 2 3 4 5 6 7 8 9

Enter Matrix Element (3 X 3) :
9 8 7 6 5 4 3 2 1

First Matrix :

1      2      3
4      5      6
7      8      9

Second Matrix :

9      8      7
6      5      4
3      2      1

Addition of Matrix :

10     10     10
10     10     10
10     10     10
```

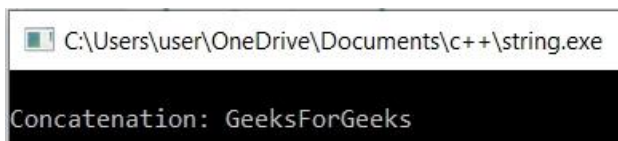
9. Write a program to concatenate the two strings using Operator Overloading

Code:

```
#include <iostream>
#include <string.h>

using namespace std;
class AddString {
public:
    char s1[25], s2[25];
    AddString(char str1[], char str2[])
    {
        strcpy(this->s1, str1);
        strcpy(this->s2, str2);
    }
    void operator+()
    {
        cout << "\nConcatenation: " << strcat(s1, s2);
    }
};
int main()
{
    char str1[] = "Geeks";
    char str2[] = "ForGeeks";
    AddString a1(str1, str2);
    +a1;
    return 0;
}
```

Output:



```
C:\Users\user\OneDrive\Documents\c++\string.exe
Concatenation: GeeksForGeeks
```

10. Write a program to calculate the area of circle, rectangle and triangle using Function Overloading

- Ø Rectangle: $\text{Area} * \text{breadth}$
- Ø Triangle: $\frac{1}{2} * \text{Area} * \text{breadth}$
- Ø Circle: $\text{Pi} * \text{Area} * \text{Area}$

Code:

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

double calculateArea(double length, double breadth) {
    return length * breadth;
}

double calculateArea(double radius) {
    return M_PI * radius * radius;
}

int main() {
    char choice;
    double area;

    cout << "Choose a shape (C for circle, R for rectangle, T for triangle): ";
    cin >> choice;

    if (choice == 'C' || choice == 'c') {
        double radius;
        cout << "Enter the radius of the circle: ";
        cin >> radius;
        area = calculateArea(radius);
        cout << "Area of the circle: " << area << endl;
    } else if (choice == 'R' || choice == 'r') {
        double length, breadth;
        cout << "Enter the length of the rectangle: ";
```

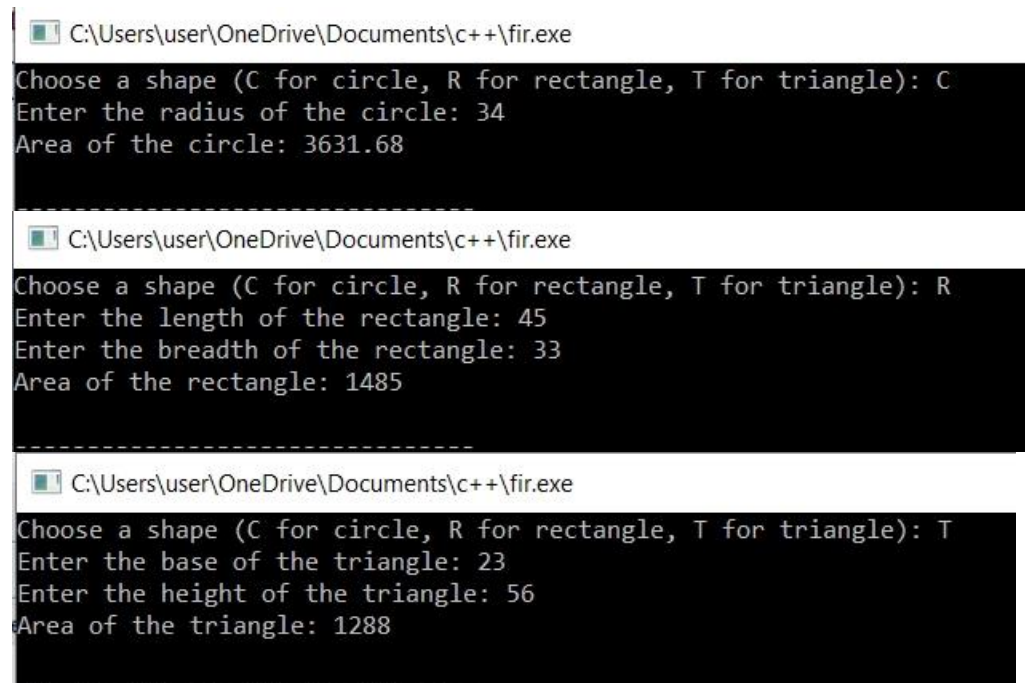
```

    cin >> length;
    cout << "Enter the breadth of the rectangle: ";
    cin >> breadth;
    area = calculateArea(length, breadth);
    cout << "Area of the rectangle: " << area << endl;
} else if (choice == 'T' || choice == 't') {
    double base, height;
    cout << "Enter the base of the triangle: ";
    cin >> base;
    cout << "Enter the height of the triangle: ";
    cin >> height;
    area = calculateArea(base, height);
    cout << "Area of the triangle: " << area << endl;
} else {
    cout << "Invalid choice. Please choose C, R, or T." << endl;
}

return 0;
}

```

Output:



The image displays three separate screenshots of a Windows command prompt window running a C++ program named 'fir.exe' located at 'C:\Users\user\OneDrive\Documents\c++\fir.exe'.

First Screenshot: The program prompts the user to 'Choose a shape (C for circle, R for rectangle, T for triangle):'. The user enters 'C'. The program then prompts 'Enter the radius of the circle:'. The user enters '34'. The program outputs 'Area of the circle: 3631.68'.

Second Screenshot: The program prompts the user to 'Choose a shape (C for circle, R for rectangle, T for triangle):'. The user enters 'R'. The program then prompts 'Enter the length of the rectangle:'. The user enters '45'. The program then prompts 'Enter the breadth of the rectangle:'. The user enters '33'. The program outputs 'Area of the rectangle: 1485'.

Third Screenshot: The program prompts the user to 'Choose a shape (C for circle, R for rectangle, T for triangle):'. The user enters 'T'. The program then prompts 'Enter the base of the triangle:'. The user enters '23'. The program then prompts 'Enter the height of the triangle:'. The user enters '56'. The program outputs 'Area of the triangle: 1288'.

11. Write a program to swap the two numbers using friend function without using third variable.

Code:

```
#include <iostream>

using namespace std;

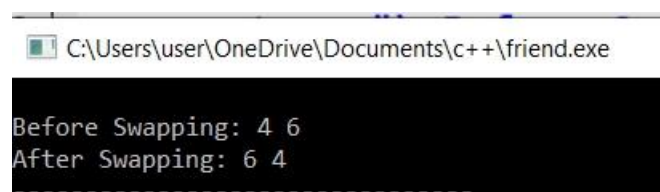
class Swap {
    int temp, a, b;

public:
    Swap(int a, int b)
    {
        this->a = a;
        this->b = b;
    }
    friend void swap(Swap&);
};

void swap(Swap& s1)
{
    cout << "\nBefore Swapping: " << s1.a << " " << s1.b;
    s1.temp = s1.a;
    s1.a = s1.b;
    s1.b = s1.temp;
    cout << "\nAfter Swapping: " << s1.a << " " << s1.b;
}

int main()
{
    Swap s(4, 6);
    swap(s);
    return 0;
}
```

Output:



```
C:\Users\user\OneDrive\Documents\c++\friend.exe
Before Swapping: 4 6
After Swapping: 6 4
-----
```

12. Write a program to find the max number from given two numbers using friend function

Code:

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

class Test {
private:
    int x, y;
public:

    void input() {
        cout << "Enter two numbers:";
        cin >> x>>y;
    }

    friend void find(Test t);
};

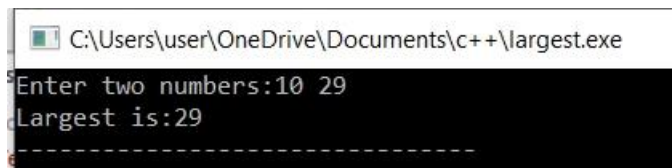
void find(Test t) {
    if (t.x > t.y) {
        cout << "Largest is:" << t.x;
    } else {
        cout << "Largest is:" << t.y;
    }
}

int main() {

    Test t;
    t.input();
    find(t);

    return 0;
}
```


Output:



```
C:\Users\user\OneDrive\Documents\c++\largest.exe
Enter two numbers:10 29
Largest is:29
-----
```

The image shows a screenshot of a Windows command prompt window. The title bar at the top reads "C:\Users\user\OneDrive\Documents\c++\largest.exe". The command prompt displays the text "Enter two numbers:10 29" on the first line and "Largest is:29" on the second line. A dashed line is visible on the third line.