**LAB 13:  
PRACTICE TASK 1:  
CODE:**

#include <iostream>

#include <cmath>

class Calculator {

public:

// Static methods for basic arithmetic operations

static double sum(double a, double b) {

return a + b;

}

static double multiply(double a, double b) {

return a \* b;

}

static double divide(double a, double b) {

if (b != 0.0) {

return a / b;

}

else {

std::cerr << "Error: Division by zero." << std::endl;

return std::numeric\_limits<double>::quiet\_NaN(); // Not a Number (NaN) to indicate an error

}

}

static double modulus(double a, double b) {

if (b != 0.0) {

return std::fmod(a, b);

}

else {

std::cerr << "Error: Modulus by zero." << std::endl;

return std::numeric\_limits<double>::quiet\_NaN(); // NaN to indicate an error

}

}

// Static methods for trigonometric functions

static double sin(double angle) {

return std::sin(angle);

}

static double cos(double angle) {

return std::cos(angle);

}

static double tan(double angle) {

return std::tan(angle);

}

};

int main() {

using namespace std;

// Example usage without creating an object

cout << "Sum: " << Calculator::sum(5.0, 3.0) << endl;

cout << "Product: " << Calculator::multiply(5.0, 3.0) << endl;

cout << "Division: " << Calculator::divide(10.0, 2.0) << endl;

cout << "Modulus: " << Calculator::modulus(10.0, 3.0) << endl;

cout << "Sin: " << Calculator::sin(30.0) << endl;

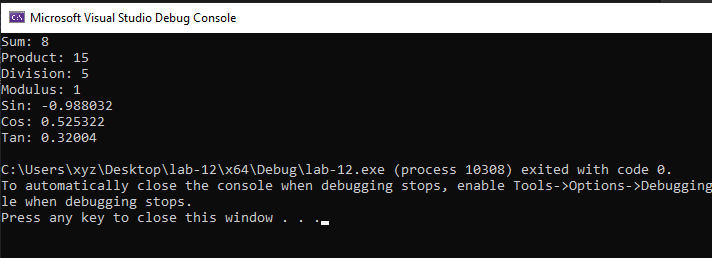
cout << "Cos: " << Calculator::cos(45.0) << endl;

cout << "Tan: " << Calculator::tan(60.0) << endl;

return 0;

}

**OUTPUT:**

****

**PRACTICE 2:**

**CODE:**

#include <iostream>

#include <string>

class Employee {

private:

std::string name;

int emp\_id;

double salary;

static int count; // Static variable to count total number of objects created

public:

// Constructors

Employee() : emp\_id(0), salary(0.0) {

count++; // Increment count when an object is created

}

Employee(const std::string& emp\_name, int id, double emp\_salary) : name(emp\_name), emp\_id(id), salary(emp\_salary) {

count++; // Increment count when an object is created

}

// Setters and Getters

void setName(const std::string& emp\_name) {

name = emp\_name;

}

void setEmpId(int id) {

emp\_id = id;

}

void setSalary(double emp\_salary) {

salary = emp\_salary;

}

std::string getName() const {

return name;

}

int getEmpId() const {

return emp\_id;

}

double getSalary() const {

return salary;

}

// Static method to get the total count of Employee objects

static int getCount() {

return count;

}

};

// Initializing the static variable count

int Employee::count = 0;

int main() {

using namespace std;

// Creating Employee objects and using setters to input data

Employee emp1;

emp1.setName("John Doe");

emp1.setEmpId(101);

emp1.setSalary(50000.0);

Employee emp2("Jane Smith", 102, 60000.0);

// Displaying employee information using getters

cout << "Employee 1 details:" << endl;

cout << "Name: " << emp1.getName() << endl;

cout << "Employee ID: " << emp1.getEmpId() << endl;

cout << "Salary: $" << emp1.getSalary() << endl;

cout << "\nEmployee 2 details:" << endl;

cout << "Name: " << emp2.getName() << endl;

cout << "Employee ID: " << emp2.getEmpId() << endl;

cout << "Salary: $" << emp2.getSalary() << endl;

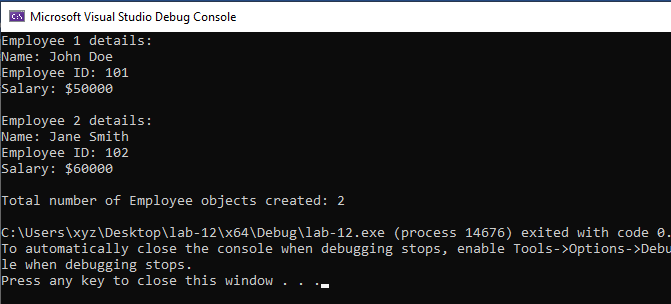
// Displaying total count of Employee objects

cout << "\nTotal number of Employee objects created: " << Employee::getCount() << endl;

return 0;

}

**OUTPUT:**

****

**LAB 14:  
PRACTICE TASK 1:  
CODE:**

#include <iostream>

class Square; // Forward declaration

class Circle {

private:

double radius;

public:

Circle(double r) : radius(r) {}

// Friend function to calculate the perimeter of the square

friend double calculatePerimeter(const Square& square);

// Friend class to calculate the area of both the circle and the square

friend class Area;

};

class Square {

private:

double length;

public:

Square(double len) : length(len) {}

// Friend function to calculate the perimeter of the square

friend double calculatePerimeter(const Square& square);

// Friend class to calculate the area of both the circle and the square

friend class Area;

};

// Friend function to calculate the perimeter of the square

double calculatePerimeter(const Square& square) {

return 4 \* square.length;

}

// Friend class to calculate the area of both the circle and the square

class Area {

public:

// Function to calculate the area of the circle

static double calculateCircleArea(const Circle& circle) {

return 3.141592653589793 \* circle.radius \* circle.radius;

}

// Function to calculate the area of the square

static double calculateSquareArea(const Square& square) {

return square.length \* square.length;

}

};

int main() {

using namespace std;

// Creating objects of Circle and Square classes

Circle circle(5.0);

Square square(4.0);

// Calculating and displaying perimeter of the square

double squarePerimeter = calculatePerimeter(square);

cout << "Perimeter of the square: " << squarePerimeter << endl;

// Calculating and displaying area of the circle

double circleArea = Area::calculateCircleArea(circle);

cout << "Area of the circle: " << circleArea << endl;

// Calculating and displaying area of the square

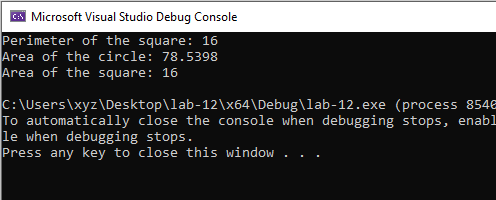
double squareArea = Area::calculateSquareArea(square);

cout << "Area of the square: " << squareArea << endl;

return 0;

}

**OUTPUT:**

****

**Practice 2:**

**Code:**

#include <iostream>

// Forward declaration of Calculation class

class Calculation;

// Data class definition

class Data {

private:

float a;

float b;

public:

Data(float val\_a, float val\_b) : a(val\_a), b(val\_b) {}

// Friend function to display data members of the Data class

friend void display(const Data& data);

// Friend class Calculation

friend class Calculation;

};

// Friend function to display data members of the Data class

void display(const Data& data) {

std::cout << "Data members: a = " << data.a << ", b = " << data.b << std::endl;

}

// Calculation class definition

class Calculation {

public:

// Function to compute the Square of sums: (a + b)^2

static float squareOfSums(const Data& data) {

return data.a \* data.a + data.b \* data.b + 2 \* data.a \* data.b;

}

// Function to compute the Square of difference: (a - b)^2

static float squareOfDifference(const Data& data) {

return data.a \* data.a + data.b \* data.b - 2 \* data.a \* data.b;

}

};

int main() {

using namespace std;

// Creating an object of the Data class

Data dataObject(5.0, 3.0);

// Displaying data members using the friend function

display(dataObject);

// Using the Calculation class to perform operations on Data class

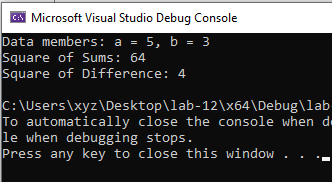
cout << "Square of Sums: " << Calculation::squareOfSums(dataObject) << endl;

cout << "Square of Difference: " << Calculation::squareOfDifference(dataObject) << endl;

return 0;

}

**Output:**

****

**Practice task 3:**

**Code:**

#include <iostream>

class Equation {

private:

int a;

int b;

int result1;

int result2;

int result3;

public:

// Constructors

Equation() : a(0), b(0), result1(0), result2(0), result3(0) {}

Equation(int val\_a, int val\_b) : a(val\_a), b(val\_b), result1(0), result2(0), result3(0) {}

// Setters and Getters

void setA(int val\_a) {

a = val\_a;

}

int getA() const {

return a;

}

void setB(int val\_b) {

b = val\_b;

}

int getB() const {

return b;

}

void setResult1(int res1) {

result1 = res1;

}

int getResult1() const {

return result1;

}

void setResult2(int res2) {

result2 = res2;

}

int getResult2() const {

return result2;

}

void setResult3(int res3) {

result3 = res3;

}

int getResult3() const {

return result3;

}

// Friend function to determine the results of all three equations

friend void calculateResults(Equation& eq);

// Friend function to display the results

friend void displayResults(const Equation& eq);

};

// Friend function to determine the results of all three equations

void calculateResults(Equation& eq) {

// Perform computations and set results

eq.setResult1(eq.getA() + eq.getB());

eq.setResult2(eq.getA() \* eq.getB());

eq.setResult3(eq.getA() - eq.getB());

}

// Friend function to display the results

void displayResults(const Equation& eq) {

using namespace std;

cout << "Result 1: " << eq.getResult1() << endl;

cout << "Result 2: " << eq.getResult2() << endl;

cout << "Result 3: " << eq.getResult3() << endl;

}

int main() {

using namespace std;

// Creating an object of the Equation class

Equation eq(5, 3);

// Using setters and getters to modify and retrieve values

eq.setA(10);

eq.setB(2);

// Calculating results using the friend function

calculateResults(eq);

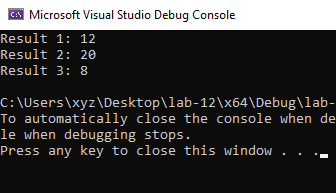
// Displaying results using the friend function

displayResults(eq);

return 0;

}

**Output:**

****