**Object Oriented Programming**

**Fall 2022**

|  |  |  |
| --- | --- | --- |
| **Lab 08** | |  |
| **Topic** | **String class and string operators overloading, static data members, static member functions, constant data members and constant member functions** |

**Task#1:** Write a class String which have the following two members (char \*) and int len. Write appropriate Constructors for it. You are also required to overload the ‘+’ operator to concatenate the two Strings member

**Task#2:** Write a program that allows two players to play the tic-tac-toe game. Your program must contain the **class ticTacToe** to implement a **ticTacToe** object. Include a 3-by-3 two-dimensional array, as a **private** member variable, to create the board. If needed, include additional member variables. Some of the operations on a **ticTacToe** object are printing the current board, getting a move, checking if a move is valid, and determining the winner after each move. Add additional operations as needed.

**Task#3:** The equation of a line in standard form is ax + by = c, where in both a and b cannot be zero, and a, b, and c are real numbers. If b ≠ 0, then –a/b is the slope of the line. If a = 0, then it is a horizontal line, and if b = 0, then it is a vertical line. The slope of a vertical line is undefined. Two lines are parallel if they have the same slope or both are vertical lines. Two lines are perpendicular if either one of the lines is horizontal and the other is vertical or the product of their slopes is –1. Design the **class lineType** to store a line. To store a line, you need to store the values of a (coefficient of x), b (coefficient of y), and c. Your **class** must contain the following operations:

* 1. If a line is nonvertical, then determine its slope.
  2. Determine if two lines are equal. (Two lines a*1*x +b1y = c1 and a2x + b2 y = c2 are equal if either a1 = a2, b1 =b2, and c1 =c2 or a1 =ka2, b1 =kb2, and c1 =kc2 for some real number k.)
  3. Determine if two lines are parallel.
  4. Determine if two lines are perpendicular.
  5. If two lines are not parallel, then find the point of intersection.

Add appropriate constructors to initialize variables of **lineType**. Also write a

program to test your **class**.

**Task#4:** Defines the **class clockType** to implement time in a program. Add functions to this **class** so that a program that uses this **class** can set only the hours, minutes, or seconds and retrieve only the hours, minutes, or seconds. Make the functions that retrieve hours, minutes, and seconds as inline. Also write a program to test your **class**.

**Task#5:** Design and implement a **class dayType** that implements the day of the week in a program. The **class dayType** should store the day, such as **Sun** for Sunday. The program should be able to perform the following operations on an object of type **dayType**:

1. Set the day.
2. Print the day.
3. Return the day.
4. Return the next day.
5. Return the previous day.
6. Calculate and return the day by adding certain days to the current day. For example, if the current day is Monday and we add 4 days, the day to be returned is Friday. Similarly, if today is Tuesday and we add 13 days, the day to be returned is Monday.
7. Add the appropriate constructors.

**Task#6:**

**Run the given below code about static data members and check the output and functionality of static datamember and if we remove the keyword static from the data member**  static int objectCount; then what will be the output

#include <iostream>

using namespace std;

class Box {

public:

static int objectCount;

// Constructor definition

Box(double l = 2.0, double b = 2.0, double h = 2.0) {

cout <<"Constructor called." << endl;

length = l;

breadth = b;

height = h;

// Increase every time object is created

objectCount++;

}

double Volume() {

return length \* breadth \* height;

}

static int getCount() {

return objectCount;

}

private:

double length; // Length of a box

double breadth; // Breadth of a box

double height; // Height of a box

};

// Initialize static member of class Box

int Box::objectCount = 0;

int main(void) {

// Print total number of objects before creating object.

cout << "Inital Stage Count: " << Box::getCount() << endl;

Box Box1(3.3, 1.2, 1.5); // Declare box1

Box Box2(8.5, 6.0, 2.0); // Declare box2

// Print total number of objects after creating object.

cout << "Final Stage Count: " << Box::getCount() << endl;

return 0;

}

**Task 7**

**Run the given below code about static data members and check the output and functionality of static function that how we are calling the member function of the class without creating the object of the class OR without using the object of the class we are calling the member function of the class.**

#include <iostream>

using namespace std;

**class** Example{

**static** **int** Number;

**int** n;

**public**:

**void** set\_n(){

        n = ++Number;

    }

**void** show\_n(){

        cout<<"value of n = "<<n<<endl;

    }

**static** **void** show\_Number(){

        cout<<"value of Number = "<<Number<<endl;

    }

};

**int** Example:: Number;

**int** main()

{

    Example example1, example2;

    example1.set\_n();

    example2.set\_n();

    example1.show\_n();

    example2.show\_n();

    Example::show\_Number();

**return** 0;

}

**Task 8**

**Run the given below code and check the functionality of the constant member function and also try to change the value of x in the constant function and check that what happens?**

#include<iostream>

**using** **namespace** std;

**class** Demo

{

**int** x;

**public**:

**void** set\_data(**int**);

**int** get\_data() **const**;

};

**void** Demo::set\_data(**int** a)

{

    x=a;

}

**int** Demo::get\_data() **const**

 {

**return** x;

 }

main()

{

    Demo d;

    d.set\_data(10);

    cout<<endl<<d.get\_data();

**return** 0;

}