CSC 240

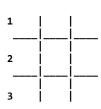
Lab 1

Option 1:

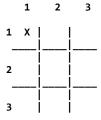
Complete the implementation of the Tic-Tac-Toe game. Use the provided header file: ticTacToe.h. Also, use the provided driver file for testing purposes: ticTacToeDriver.cpp. These files can be found in the Extra Files folder under Content in D2L.

Use the following code for your play function:

```
void ticTacToe::play()
{
    bool done = false;
    char player = 'X';
    displayBoard();
    while (!done)
    {
        done = getXOMove(player);
        if (player == 'X')
            player = '0';
        else
            player = 'X';
    }
}
Here is a demo of a game:
  1 2 3
```



Player X enter move: 1 1



Player O enter move: 1 2

1 2 3

1 X | 0 | ______ 2 | _____ 3

Player X enter move: 2 2

1 2 3

1 X 0 | 2 X | 3

Player O enter move: 3 3

1 2 3

Player X enter move: 2 1

1 2 3

Player O enter move: 3 1

1 2 3

```
Player X enter move: 3 1
Player X enter move: 3 2
       2
            3
1 X | 0 |
2 X |
       Χ
3 0 | X | 0
Player O enter move: 1 3
  1
       2
            3
1 X |
2 X |
       Χ
3 0 | X | 0
Player X enter move: 2 3
  1
       2
            3
1 X |
       0 |
           0
2 X |
       χį
3 0 | X |
Player X wins!
```

Option 2:

Complete the implementation of pointType and circleType. Use the provided header files: **pointType.h** and **circleType.h**. Also, write a driver program for testing the implemented functionality of the circleType class. Be sure to test all of the functions listed here: print, setRadius, getRadius, getCircumference, getArea, circleType.

Add and implement the following function to the circleType class: setCenter.

Use the following signature for the setCenter function:

```
void setCenter(pointType&);
Use the following implemented functions for your circleType class:
circleType::circleType(double x, double y, double r) : pointType(x, y){
            radius = r;
}
```

Notice the use of the initialization of the pointType for the circleType constructor. This notation is referred to as an initialization list.

```
void circleType::print() const{
    cout << "center: ";
    pointType::print();
    cout << "radius = " << radius << endl;
}</pre>
```

Create a new class called Sphere that inherits the circleType class and has the following properties: x-coordinate, y-coordinate, z-coordinate, and radius. Where the radius and x and y-coordinates are used to initialize the circleType base class, and the z-coordinate is a data member of the Sphere. Implement a member function called volume() that will return the volume of the Sphere object. Complete the implementation of the Sphere class and test it in the same driver used to test the circleType class.

Option 2 is useful for reviewing the concept of inheritance where one class (the derived class) inherits properties of another class (the base class). In this case, the pointType class is the base class and the circleType class is the derived class.