Evan Nguyen
Joseph Guzman
CECS 275
4/10/2022
Lab 5

The report is formatted where the outputs of both problems are shown first, then the code afterwards.

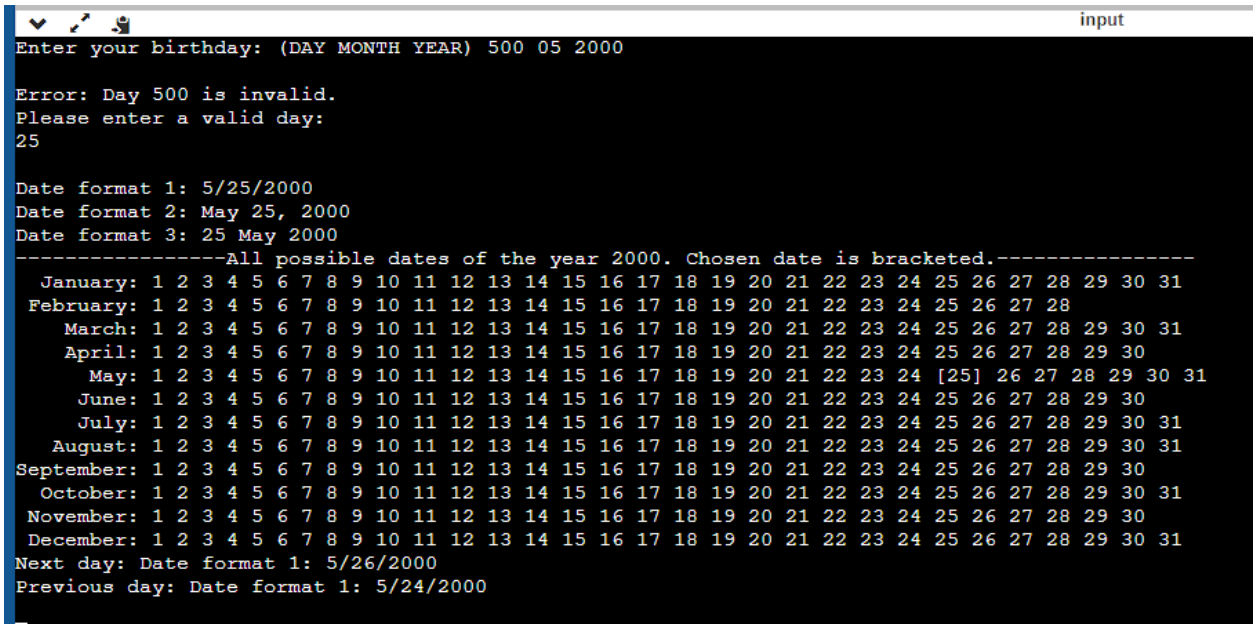
Problem 1 Output:

input

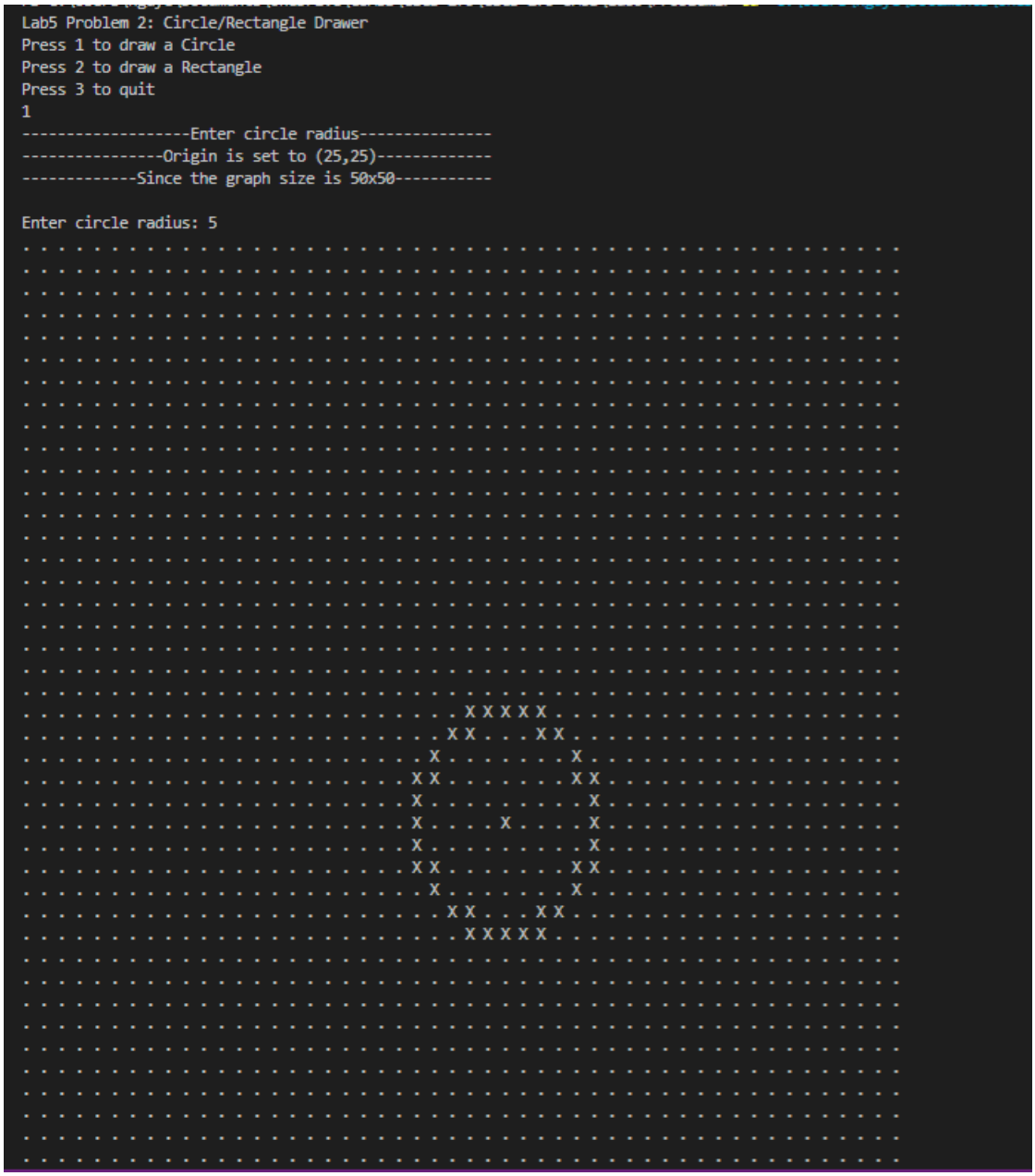
```
Enter your birthday: (DAY MONTH YEAR) 29 02 2016

Date format 1: 2/29/2016
Date format 2: February 29, 2016
Date format 3: 29 February 2016
-----------------All possible dates of the year 2016. Chosen date is bracketed.----------------
  January: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
 February: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 [29]
    March: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
    April: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
      May: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
     June: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
     July: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
   August: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
September: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
  October: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
 November: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
 December: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Next day: Date format 1: 3/1/2016
Previous day: Date format 1: 2/28/2016
```

input

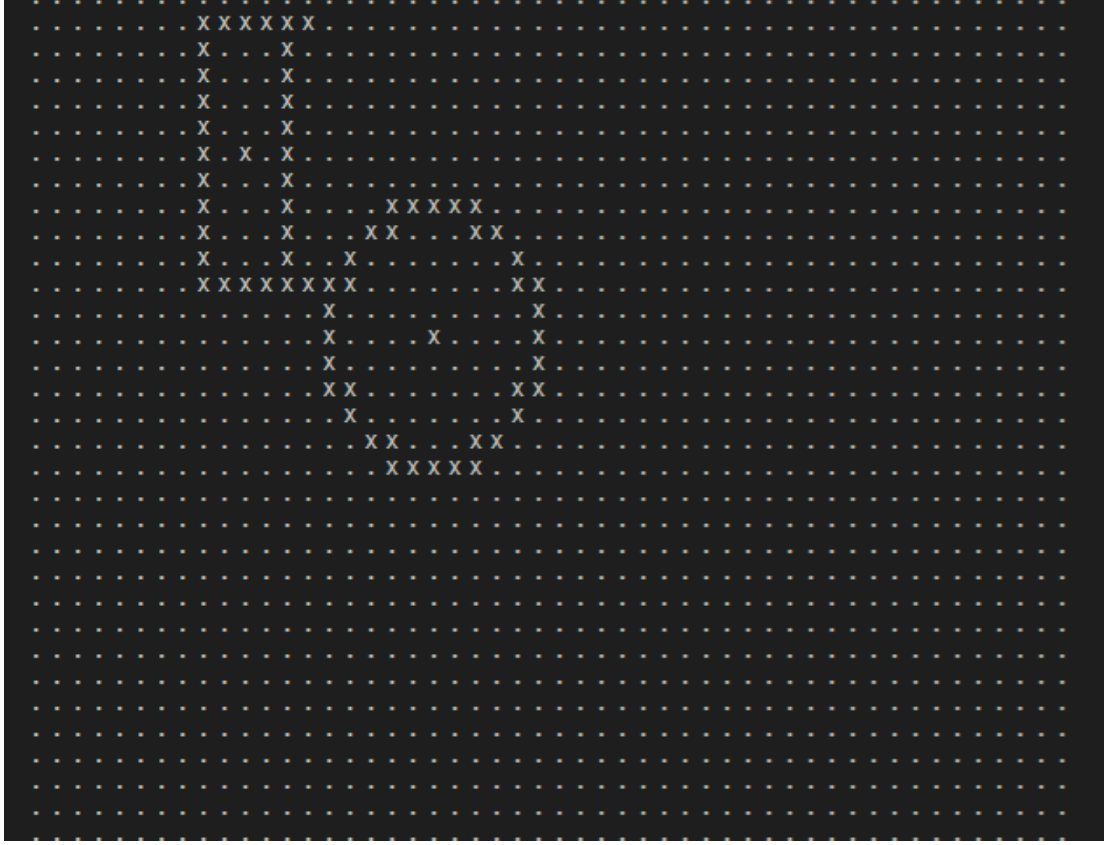
```
Enter your birthday: (DAY MONTH YEAR) 01 02 2016

Date format 1: 2/1/2016
Date format 2: February 1, 2016
Date format 3: 1 February 2016
-----------------All possible dates of the year 2016. Chosen date is bracketed.----------------
  January: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
 February: [1] 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
    March: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
    April: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
      May: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
     June: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
     July: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
   August: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
September: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
  October: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
 November: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
 December: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Next day: Date format 1: 2/2/2016
Previous day: Date format 1: 1/31/2016
```


input
Enter your birthday: (DAY MONTH YEAR) 500 05 2000
Error: Day 500 is invalid.
Please enter a valid day:
25
Date format 1: 5/25/2000
Date format 2: May 25, 2000
Date format 3: 25 May 2000
----------All possible dates of the year 2000. Chosen date is bracketed.----------
January: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
February: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
March: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
April: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
May: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 [25] 26 27 28 29 30 31
June: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
July: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
August: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
September: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
October: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
November: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
December: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Next day: Date format 1: 5/26/2000
Previous day: Date format 1: 5/24/2000

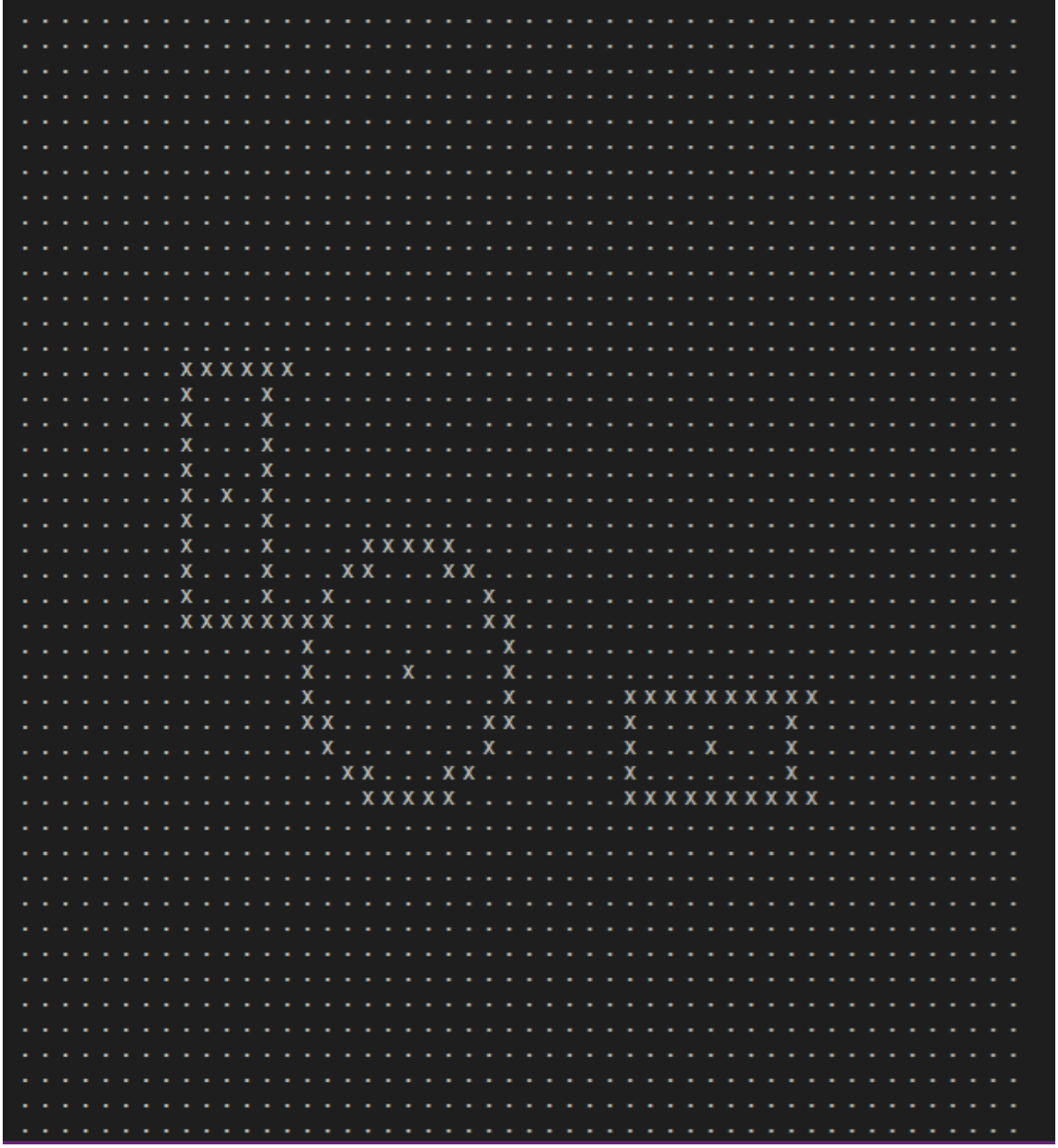
Problem 2 Output:



```
------------------Enter Height/Width----------------
---------------Origin is set to (25,25)-------------
------------Since the graph size is 50x50-----------

Enter height: 10

Enter width: 5
```



```
------------------Enter Height/Width----------------
---------------Origin is set to (25,25)-------------
------------Since the graph size is 50x50-----------

Enter height: 4

Enter width: 9
```



Problem 1 Code:
(main.cpp)

main.cpp | Date.h | Date.cpp

```cpp
#include "Date.h"
#include <iostream>

using  namespace std;
int main()
{
    int month = 0;
    int day = 0;
    int year = 0;
    bool isDone = false;


    cout << "Enter your birthday: (DAY MONTH YEAR) ";
    cin >> day >> month >> year;
    cout << "\n";

    while (!isDone)
    {
        try
        {
            //Date dateObj(day, month, year);

            //dateObj(day, month, year);
            Date dateObj;
            dateObj.setMonth(month);
            dateObj.setDay(day);
            dateObj.setYear(year);

            dateObj.getToString1();
            dateObj.getToString2();
            dateObj.getToString3();

            dateObj.listAllDates();
            dateObj++;
            dateObj--;

```

```
36
37              isDone = true;
38          }
39          catch (Date::InvalidDay dayVal)
40          {
41              cout << "Error: Day " << dayVal.getInvalidDay() << " is invalid.\n";
42              cout << "Please enter a valid day: \n";
43              cin >> day;
44          }
45          catch (Date::InvalidMonth monthVal)
46          {
47              cout << "Error: Month " << monthVal.getInvalidMonth() << " does not exist.\n";
48              cout << "Please enter a valid month: \n";
49              cin >> month;
50          }
51          catch (Date::InvalidYear yearVal)
52          {
53              cout << "Error: Year " << yearVal.getInvalidYear() << " does not exist.\n";
54              cout << "Please enter a valid year: \n";
55              cin >> year;
56          }
57          cout << "\n";
58      }
59
60
61      cin.ignore();
62      cin.get();
63
64
65      return 0;
66  }
```

Date.h

main.cpp | Date.h | Date.cpp

```
#ifndef DATE_H
#define DATE_H

#include <string>

class Date
{
    private:
        int month;
        int day;
        int year;

    public:

        class InvalidDay
        {
            private:
                int invalidDay;

            public:
                InvalidDay(int dayVal)
                { invalidDay = dayVal; }

                int getInvalidDay() const
                { return invalidDay; }
        };

        class InvalidMonth
        {
            private:
                int invalidMonth;

            public:
                InvalidMonth(int monthVal)
                { invalidMonth = monthVal; }

```

```
36
37                  int getInvalidMonth() const
38                  { return invalidMonth; }
39          };
40
41          class InvalidYear
42          {
43              private:
44                  int invalidYear;
45
46              public:
47                  InvalidYear(int yearVal)
48                  { invalidYear = yearVal; }
49
50                  int getInvalidYear() const
51                  { return invalidYear; }
52          };
53
54          Date();
55          Date(int aDay, int aMonth, int aYear);
56          /* const Date Operator++(const Date& otherDate);*/
57          Date operator++(int aDay);
58          Date operator--(int aDay);
59
60          // Mutator functions
61          bool isLeapYear();
62          void setMonth(int aMonth);
63          void setDay(int aDay);
64          void setYear(int aYear);
65          void listAllDates();
66
67          // Accessor functions
68          void getToString1() const;
69          void getToString2() const;
70          void getToString3() const;
71  };
```

Date.cpp

```cpp
#include "Date.h"
#include <iostream>
#include <array>
#include <iomanip>
using  namespace std;

array<int, 13> monthDays { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
const array<string, 13> monthNames { "", "January", "February", "March", "April",
                                     "May", "June", "July", "August", "September",
                                     "October", "November", "December" };
enum Months { January = 1, February, March, April, May, June, July,
              August, September, October, November, December };


Date::Date():Date(1, 1, 1970){

}
Date::Date(int aDay, int aMonth, int aYear){
    day = aDay;
    month = aMonth;
    year = aYear;
}

void Date::setMonth(int aMonth)
{
    if (aMonth >= January && aMonth <= December)
    {
        month = aMonth;
    }
    else
    {
        throw InvalidMonth(aMonth);
    }
}
```

main.cpp | Date.h | Date.cpp

```
36  void Date::setDay(int aDay)
37  {
38      if (month == February && isLeapYear() && aDay == 29)
39      {
40          day = aDay;
41          monthDays[2] = 29;
42      }
43      else if (aDay < 1 || aDay > monthDays[month])
44      {
45          throw InvalidDay(aDay);
46      }
47      else
48      {
49          day = aDay;
50      }
51  }
52
53  void Date::setYear(int aYear)
54  {
55      if (aYear >= 1 && aYear <= 2023)
56      {
57          this->year = aYear;
58      }
59      else
60      {
61          throw InvalidYear(aYear);
62      }
63  }
64
65  bool Date::isLeapYear()
66  {
67      //cout << year << endl;
68      if (year % 4 == 0)
69      {
70          return true;
71      }
```

```
    else if (year % 100 != 0)
    {
        return true;
    }
    else if (year % 400 == 0)
    {
        return true;
    }
    else
    {
        return false;
    }
}

void Date::listAllDates() {
    // iterate thru the months and output each date
    cout << "-----------------All possible dates of the year " << year <<
            ". Chosen date is bracketed.----------------" << endl;
    for (int monthInt = January; monthInt < December + 1; monthInt++){
        cout << setw(9) << monthNames[monthInt] << ": ";
        for (int i = 1; i < monthDays[monthInt] + 1; i++) {
            // check if we're in the day and month we set, put it in brackets
            if ((i == day) && (monthInt == month)){
                cout << "[" << i << "] ";
            } else {
            cout << i << " ";
            }
        }
        cout << endl;
    }
}

Date Date::operator++(int aDay){
    Date temp;
    // assign temp to this->day and increase it
```

```
Date Date::operator++(int aDay){
    Date temp;
    // assign temp to this->day and increase it
    temp.day = this->day+1;
    //day++;

    if (temp.day > monthDays[month]){
        month = month+1;
        day = 1;
    } else {
        day++;
    }

    cout << "Next day: ";
    getToString1();

    return temp;
}

Date Date::operator--(int aDay){
    Date temp;
    // assign temp to this->day and decrease it
    temp.day = this->day;
    day--;
    day--;
    // if leap year then make sure it works
    if (day < 1){
        month = month-1;
        day = monthDays[month];
        if (day == 29 && monthDays[month] == 29){
            day--;
        }
    } else {

    } else {

    }
    cout << "Previous day: ";
    getToString1();
    return temp;
}

void Date::getToString1() const
{
    cout << "Date format 1: " << month << "/" << day << "/" << year << "\n";
}

void Date::getToString2() const
{
    cout << "Date format 2: " << monthNames[month] << " " << day << ", " << year << "\n";
}

void Date::getToString3() const
{
    cout << "Date format 3: " << day << " " << monthNames[month] << " " << year << "\n";
}
```

Problem 2 Code:
(main.cpp)

main.cpp

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab5 > Problem2 > main.

```cpp
/*
 * Answer to #2 on Lab 5
 * CECS 275 - Spring 2022
 * @author Evan Nguyen
 * @author Joseph Guzman
 * @version 1.0.0
 *
 */

#include <iostream>
#include <stdlib.h>

#include "Shape.h"
#include "Shape.cpp"
#include "Circle.h"
#include "Circle.cpp"
#include "Rectangle.h"
#include "Rectangle.cpp"

using namespace std;

int main() {

    int userInput;

    int centerX;
    int centerY;

    int circRadius;
    int rectH;
    int rectW;

    while(true){
        cout << "Lab5 Problem 2: Circle/Rectangle Drawer" << endl;
        cout << "Press 1 to draw a Circle" << endl;
        cout << "Press 2 to draw a Rectangle" << endl;
        cout << "Press 3 to quit" << endl;
        cin >> userInput;
        // randomized seed
        int centerX = rand() % 25 + 10;
        int centerY = rand() % 25 + 10;
        if (userInput == 1) {
            cout << "------------------Enter circle radius---------------\n";
            cout << "---------------Origin is set to (25,25)-------------\n";
            cout << "-------------Since the graph size is 50x50-----------\n";
            cout << "\nEnter circle radius: ";
            cin >> circRadius;
```

```
            cin >> circRadius;

            Circle Circ(centerX, centerY, circRadius);
            //cout << "Circle Area: " << Circ.calcArea() << endl;
            Circ.draw();
        } else if (userInput == 2) {
            cout << "-------------------Enter Height/Width----------------\n";
            cout << "----------------Origin is set to (25,25)------------\n";
            cout << "-------------Since the graph size is 50x50-----------\n";
            cout << "\nEnter height: ";
            cin >> rectH;
            cout << "\nEnter width: ";
            cin >> rectW;

            Rectangle Rect(centerX, centerY, rectW, rectH);
            //cout << "Rectangle Area: " << Rect.calcArea() << endl;
            Rect.draw();
        } else if (userInput == 3){
            return 0;
        }

    }
    return 0;
}
```

Shape.h

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab5 > Problem2 > C Shape.h > ...

```
1   #ifndef SHAPE_H
2   #define SHAPE_H
3
4   // To create an abstract class,
5   // declare at least one pure virtual memebr function
6   // Abstract base class is not instantiated, but just a base for other classes
7
8   // To be overrided by a derived class Circle and Rectangle
9
10  class Shape {
11      private:
12          double area;
13
14      public:
15          // Constructors
16          Shape();
17          Shape(double aArea);
18          ~Shape();
19          // Get/set functions
20          double getArea() const;
21          void setArea(double area);
22
23          // Virtual functions to be overridden
24          virtual double calcArea() const = 0;
25          virtual void draw() const = 0;
26  };
27
28  #endif
```

Shape.cpp

main.cpp | Shape.h | Shape.cpp | Rectangle.h

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS >

```cpp
#include "Shape.h"

using namespace std;

Shape::Shape(){
    area = 0.0;
}

Shape::Shape(double aArea){
    area = aArea;
}

double Shape::getArea() const{
    return area;
}


void Shape::setArea(double aArea){
    area = aArea;
}

// pure virtual function
double Shape::calcArea() const{
    return 0.0;
}

void Shape::draw() const{

}

Shape::~Shape(){
    //cout << "Shape destroyed" << endl;
}
```

Rectangle.h

main.cpp | Shape.h | Shape.cpp | Rectangle.h | Rectangle.cpp

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab5 > Problem2

```
#ifndef RECTANGLE_H
#define RECTANGLE_H
#include "Shape.h"

class Rectangle : public Shape {
    private:
        int width;
        int height;
        int centerX;
        int centerY;
    public:
        // Constructors
        Rectangle();
        Rectangle(int aCenterX, int aCenterY, int aWidth, int aHeight);

        // Get functions
        int getWidth() const;
        int getHeight() const;
        int getCenterX() const;
        int getCenterY() const;

        // Set functions
        void setWidth(int width);
        void setHeight(int height);
        void setCenterX(int centerX);
        void setCenterY(int centerY);

        //overridden virtual functions
        virtual double calcArea() const;
        void draw() const;
};

#endif
```

Rectangle.cpp

main.cpp | Shape.h | Shape.cpp | Rectangle.h | Rectangle.cpp

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab5 > Problem2 > R

```cpp
#include "Rectangle.h"
#include <cmath>
#include <iostream>

using namespace std;

Rectangle::Rectangle(){
    width = 0;
    height = 0;
    centerX = 0;
    centerY = 0;
}

Rectangle::Rectangle(int aCenterX, int aCenterY, int aWidth, int aHeight){
    centerX = aCenterX;
    centerY = aCenterY;
    width = aWidth;
    height = aHeight;

}

int Rectangle::getWidth() const {
    return width;
}

int Rectangle::getHeight() const {
    return height;
}

int Rectangle::getCenterX() const {
    return centerX;
}

int Rectangle::getCenterY() const {
    return centerY;
}

void Rectangle::setWidth(int aWidth){
    width = aWidth;
}
```

```cpp
void Rectangle::setHeight(int aHeight){
    height = aHeight;
}

void Rectangle::setCenterX(int aCenterX){
    centerX = aCenterX;
}

void Rectangle::setCenterY(int aCenterY){
    centerY = aCenterY;
}

//overridden virtual functions
double Rectangle::calcArea() const{
    int area = 0;
    return area = width * height;
}

void Rectangle::draw() const{
    // graph size
    int x = 100;
    int y = 100;

    // threshold = approximation of the point on the function
    // set higher = thicker border
    // set lower = thinner border (might miss some points)
    int threshold = 5;
    // if radius = 10, set to 10

    int graph[x][y] = {0}; // clear the graph
    graph[centerX][centerY] = {1}; // set the origin


    // iterate through the points
    // bia creating the corners first

    for (int x_i = 0; x_i <= width; x_i++){
        graph[centerX-height/2][centerY-width/2 + x_i] = {1};
        graph[centerX+height/2][centerY-width/2 + x_i] = {1};
    }

    for (int y_i = height; y_i >= 0; y_i--){
        graph[centerX-height/2 + y_i][centerY+width/2] = {1}; // Top right
        graph[centerX+height/2 - y_i][centerY-width/2] = {1}; // Bottom right
    }
```

```cpp
    for (int i = 0; i < 50; i++){
        for (int j = 0; j < 50; j++){
            if (graph[i][j] == 1){
                cout << "X ";
            } else {
                cout << ". ";
            }
        }
        cout << endl;
    }

    graph[x][y] = {0}; // clear the graph, doesnt work? use delete [] graph

}
```

Circle.h

main.cpp | Shape.h | Shape.cpp | Rectangle.h | Rect

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab

```
#ifndef CIRCLE_H
#define CIRCLE_H
#include "Shape.h"

class Circle : public Shape {
    private:
        int centerX; // x coord
        int centerY; // y coord
        double radius;
    public:
        // Constructors
        Circle();
        Circle(int aCenterX, int aCenterY, double aRadius);

        // Get functions
        int getCenterX() const;
        int getCenterY() const;
        double getRadius() const;

        // Set functions
        void setCenterX(int centerX);
        void setCenterY(int centerY);
        void setRadius(double radius);

        // Overridden virtual functions
        virtual double calcArea() const;
        void draw() const;
};

#endif
```

Circle.cpp

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab5 > Proble

```cpp
#include "Circle.h"
#include <cmath>
#include <iostream>

using namespace std;

const double PIE = 3.14;

Circle::Circle(){
    centerX = 0;
    centerY = 0;
    radius = 0.0;
}

Circle::Circle(int aCenterX, int aCenterY, double aRadius) {
    centerX = aCenterX;
    centerY = aCenterY;
    radius = aRadius;

}

int Circle::getCenterX() const {
    return centerX;
}

int Circle::getCenterY() const {
    return centerY;
}

double Circle::getRadius() const {
    return radius;
}
```

```cpp
void Circle::setCenterX(int aCenterX){
    centerX = aCenterX;
}

void Circle::setCenterY(int aCenterY){
    centerY = aCenterY;
}

void Circle::setRadius(double aRadius){
    radius = aRadius;
}

//overridden virtual functions
double Circle::calcArea() const {
    double area = 0.0;
    return area = PIE * pow(radius, 2);
}


void Circle::draw() const {

    // graph size
    int x = 100;
    int y = 100;

    // threshold = approximation of the point on the function
    // set higher = thicker border
    // set lower = thinner border (might miss some points)
    int threshold = 5;
    // if radius = 10, set to 10


    int graph[x][y] = {0}; // clear the graph
    graph[centerX][centerY] = {1}; //origin
```

```
67      graph[centerX][centerY] = {1}; //origin
68
69      // equation variables
70      int op1 = 0;
71      int op2 = 0;
72      int calc = 0;
73
74      // iterate through x and y of 2d
75      // and see if it satisfies the equation
76      // of a circle at the origin
77      // x^2 + y^2 = r^2
78
79      for (int y_i = 0; y_i < y; y_i++){
80          for (int x_i = 0; x_i < x; x_i++){
81
82              op1 = pow(abs(x_i - centerX), 2); // x^2
83              op2 = pow(abs(y_i - centerY), 2); // y^2
84              calc = abs(op1 + op2 - pow(radius, 2)); //x^2 + y^2 - r^2
85
86              // EX: Checking the coordinates (10,10) where origin is (50,50):
87
88              // (x-50)^2 = abs(10-50)^2 = 1600
89              // (y-50)^2 = abs(10-50)^2 = 1600
90              // assuming r = 5, r^2 = 25
91              // if 1600 + 1600 - 25 <= 5? No, so don't plot it.
92
93              // if x = 45, y = 50,
94              // x --> (45-50)^2 = abs(5)^2 = 25
95              // y --> (50-50)^2 = abs(0)^2 = 0
96              // assume r = 5, r^2 = 25
97              // if 25 + 0 - 25 <= 5? Yes, plot it.
98
99              if (calc <= threshold){
00                  graph[x_i][y_i] = 1;
01              }
02          }
03      }
04
```

```
    // cut off the rest of the coordinates bc sparkles? lol
    for (int i = 0; i < 50; i++){
        for (int j = 0; j < 50; j++){
            if (graph[i][j] == 1){
                cout << "X ";
            } else {
                cout << ". ";
            }
        }
        cout << endl;
    }

}
```