

### CSC 210 - In-class Exercise (Chapter 3) -- Use C++ language / syntax to complete the following questions

1. Convert this Java program into C++ // mark what change and to what, and also mark if any statement are unnecessary

```
1 //-----
2 // Date.java by Dale/Joyce/Weems Chapter 1
3 // Supports date objects
4 // having year, month, and day attributes.
5 //-----
6
7 public class Date
8 {
9     private int year;
10    private int month;
11    private int day;
12    public static final int MINYEAR = 1583;
13
14    // Constructor
15    public Date(int newMonth, int newDay, int newYear)
16    {
17        month = newMonth;
18        day = newDay;
19        year = newYear;
20    }
21
22    // Observers
23    public int getYear()
24    {
25        return year;
26    }
27
28    public int getMonth()
29    {
30        return month;
31    }
32
33    public int getDay()
34    {
35        return day;
36    }
37
38    public int lilian()
39    {
40        // Returns the Lilian Day Number of this date.
41        // Precondition: This Date is a valid date after
42        // 10/14/1582.
43        // Computes the number of days between 1/1/0 and
44        // this date as if no calendar
45        // reforms took place, then subtracts 578,100 so
46        // that October 15, 1582 is day 1.
```

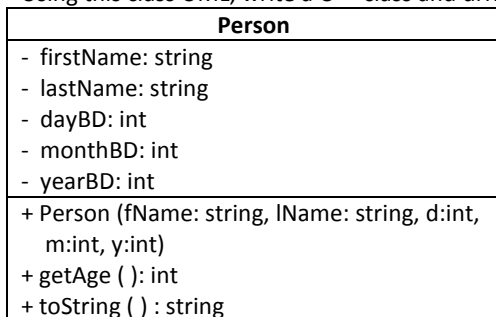
```
45
46    final int subDays = 578100; // number of calculated
47    // days from 1/1/0 to 10/14/1582
48
49    int numDays = 0;
50
51    // Add days in years.
52    numDays = year * 365;
53
54    // Add days in the months.
55    if (month <= 2)
56        numDays = numDays + (month - 1) * 31;
57    else
58        numDays = numDays + ((month - 1) * 31) -
59        ((4 * (month-1) + 27) / 10);
60
61    // Add days in the days.
62    numDays = numDays + day;
63
64    // Take care of leap years.
65    numDays = numDays + (year / 4) - (year / 100) +
66    (year / 400);
67
68    // Handle special case of leap year but not yet leap
69    // day.
70    if (month < 3)
71    {
72        if ((year % 4) == 0) numDays = numDays - 1;
73        if ((year % 100) == 0) numDays = numDays + 1;
74        if ((year % 400) == 0) numDays = numDays - 1;
75    }
76
77    // Subtract extra days up to 10/14/1582.
78    numDays = numDays - subDays;
79    return numDays;
80 }
81
82 public String toString()
83 // Returns this date as a String.
84 {
85     return(month + "/" + day + "/" + year);
86 }
87
88 // end Date class
89 }
```

```

1 //-----
2 // DaysBetween.java          by Dale/Joyce/Weems          Chapter 1
3 //
4 // Asks the user to enter two "modern" dates and then reports
5 // the number of days between the two dates.
6 //-----
7
8 import java.util.Scanner;
9
10 public class DaysBetween
11 {
12     public static void main(String[] args)
13     {
14         Scanner conIn = new Scanner(System.in);
15         int day, month, year;
16
17         System.out.println("Enter two 'modern' dates: month day year");
18         System.out.println("For example, January 12, 1954, would be: 1 12 1954");
19         System.out.println();
20         System.out.println("Modern dates occur after " + Date.MINYEAR + ".");
21         System.out.println();
22
23         System.out.println("Enter the first date:");
24         month = conIn.nextInt();
25         day = conIn.nextInt();
26         year = conIn.nextInt();
27         Date date1 = new Date(month, day, year);
28
29         System.out.println("Enter the second date:");
30         month = conIn.nextInt();
31         day = conIn.nextInt();
32         year = conIn.nextInt();
33         Date date2 = new Date(month, day, year);
34
35         if ((date1.getYear() <= Date.MINYEAR)
36             ||
37             (date2.getYear() <= Date.MINYEAR))
38             System.out.println("You entered a 'pre-modern' date.");
39         else
40         {
41             System.out.println("The number of days between");
42             System.out.print(date1);
43             System.out.print(" and ");
44             System.out.print(date2);
45             System.out.print(" is ");
46             System.out.println(Math.abs(date1.lilian() - date2.lilian()));
47         }
48     }
49 }

```

2. Using this class UML, write a C++ class and driver.



*getAge* – will calculate age based on the yearBD  
*toString*– will return a concatenation string of all attributes including age, which is calculated in *getAge* method

Driver: (1) Read each attribute value from the user;  
 (2) instantiate object using input values, and  
 (3) print on the screen all values including age of this person using *toString* method