AP Computer Science — Project 4 - Branching
Dr. Paul Bailey — Monday, August 28, 2017

Create one directory called `P4_Branching` to store the `.java` files. Please format the source code using appropriate indentation.

The purpose of this project is to practice the material from Savitch sections 3.1 - 3.3.

Program 1. Type and compile the `Tool` class as listed below.

```
import java.util.Scanner;

public class Tool
{
    private static Scanner scanner = new Scanner(System.in);

    public static String prompt(String p)
    {
        String r = "";
        System.out.print(p);
        return scanner.nextLine();
    }

    public static String piece(String s, String p, int n)
    {
        String r = "";
        String[] t = s.split(p);
        if (t.length >= n) r = t[n-1];
        return r;
    }

    public static int toInt(String s)
    {
        int r = 0;
        try
        {
            r = Integer.parseInt(s);
        }
        catch (Exception ex)
        { }
        return r;
    }
}
```

Program 2. Type and compile the `Date` class as listed below.

```
public class Date
{
    public static boolean checkMonth(int m)
    { return true; }

    public static boolean checkDay(int d)
    { return true; }

    public static boolean checkYear(int y)
    { return true; }

    public static String makeMonth(int m)
    { return "Mocktober"; }

    public static String makeDay(int d)
    { return "99th"; }

    public static String makeYear(int y)
    { return "1492"; }
}
```

Program 3. Type, compile, and run the `Program` class as listed below.

```
public class Program
{
    public static void main(String[] args)
    {
        String input = Tool.prompt("Enter a date in the format MM/DD/YYYY: ");
        int km = Tool.toInt(Tool.piece(input, "/", 1));
        int kd = Tool.toInt(Tool.piece(input, "/", 2));
        int ky = Tool.toInt(Tool.piece(input, "/", 3));

        if (!Date.checkMonth(km))
        {
            System.out.println("Invalid month");
            return;
        }
        if (!Date.checkDay(kd))
        {
            System.out.println("Invalid day");
            return;
        }
        if (!Date.checkYear(ky))
        {
            System.out.println("Invalid year");
            return;
        }

        String sm = Date.makeMonth(km);
        String sd = Date.makeDay(kd);
        String sy = Date.makeYear(ky);

        System.out.println("The " + sd + " day of " + sm + " in the year " + sy + ".");
    }
}
```

Program 4. Modify the `checkMonth`, `checkDay`, and `checkYear` methods in the `Date` class to check if the input is valid, and return `false` if the value is invalid.

- Valid months are integers between 1 and 12.
- Valid days are integers between 1 and 31.
- Valid years are integers between 0 and 99, or between 1001 and 2999.

Test this code using `Program` with several different inputs.

Program 5. Modify the `makeMonth`, `makeDay`, and `makeYear` methods in the `Date` class to convert the input into a string.

- `makeMonth` should return the name of the month. For example, `makeMonth(8)` should return the string `August`. Use a sequence of `if ... else` statements to accomplish this.
- `makeDay` should return a string such as `1st`, `2nd`, `3rd`, `4th`, ..., `21st`, ..., and so forth.
- `makeYear` should interpret 0 through 32 as 2000 through 2032, 33 through 99 as 1933 through 1999. It then converts the number into a string and return it.

Program 6. Add a method `public static String nameMonth(int m)` which behaves identically to `makeMonth`, but uses a `switch` statement instead of a sequence of `if`'s. Modify the `Program` class to test this new method.

The remainder of the programs require you to add methods to the `Date` class. Do not use any java libraries relating to dates or times to accomplish any of this functionality.

Program 7. Add a method `public static boolean isLeapYear(int y)` if the given year is a leap year. Write additional code the in `Program` class to test this new method, with output such as

```
The 22nd day of August in the year 2017.
This is not a leap year.
```

Program 8. Add a method `public static int monthLength(int m, int y)` which indicates the number of days in the given month. Correctly account for leap years. Write additional code the in `Program` class to test this new method, with output such as

```
The 22nd day of August in the year 2017.
This month has 31 days.
This is not a leap year.
```

Program 9. Add a method `public static int internalDate(int m, int d, int y)` which returns the number of days between January 1, 1970, and the given date. Correctly account for leap years. Write additional code the in `Program` class to test this new method, with output such as

```
The 22nd day of August in the year 2017.
The internal date is 17400.
The month has 31 days.
The year is not a leap year.
```

Program 10. Add a method `public static String externalDate(int n)` which takes the internal date `n` and returns it as a string in the format `MM/DD/YYYY`. Write additional code the in `Program` class to test this new method, with output such as

```
The 22nd day of August in the year 2017.
The internal date is 17400.
The external date is 08/22/2017.
The month has 31 days.
The year is not a leap year.
```

Program 11. Add a method `public static int getDOW(int n)` which takes an internal date `n` and returns the corresponding day of the week as an integer, where 0 means Sunday, 1 means Monday, ..., 6 means Saturday. Write additional code the in `Program` class to test this new method, with output such as

```
The 22nd day of August in the year 2017.
The internal day of the week is 2.
The internal date is 17400.
The external date is 08/22/2017.
The month has 31 days.
The year is not a leap year.
```

Program 12. Add a method `public static String nameDOW(int w)` which takes an internal day of the week (that is, 0, 1, ..., 6) and returns the appropriate string `Sunday`, `Monday`, ..., `Saturday`. Use a `switch` statement to accomplish this. Report this external day of the week on the first line, taking care to make sure that the tense is correct (past, present, future), with output such as

```
The 22nd day of August in the year 2017 was a Tuesday.
The internal day of the week is 2.
The internal date is 17400.
The external date is 08/22/2017.
The month has 31 days.
The year is not a leap year.
```