CSE 114: Computer Science I

Homework #3

Fall 2016

Assignment Due: September 23, 2016 by 11:59 pm

Directions:

- Solve the following problems to the best of your ability.
- At the top of every file you write for this assignment, include the following information in a comment, with each item on a separate line:
  - your first and last name as they appear in Blackboard
  - your Stony Brook ID #
  - the course number (CSE 114)
  - the assignment name and number (Homework #3)

⚠ Your files, Java classes, Java methods, etc. must be named and/or defined as proscribed below. Work that does not meet the program specifications (e.g., wrong file names or wrong method names) will not be graded.

⚠ Upload your `.java` files to Blackboard by the indicated due date and time. Late work will not be accepted for grading. Work is late if it is submitted after the due date and time.

⚠ Source code that does not compile will not be graded.

⚠ Do not upload `.class` files. Such files be deleted.

⚠ Do not combine your `.java` files into a zip file, rar file or other archive. Such files will be deleted. The grader will not unpack archive files to search for your source code.

⚠ Do not include any `package` declarations in your source code unless directed to do so. Points may be deducted if your code must be edited to remove unnecessary package declarations.

Assignment Objectives

By the end of this assignment you should be able to design, code, run and test original Java programs featuring if-statements, string manipulation and basic output formatting.

Part I: Time Flies When You're Having Fun (3 points)

Filename(s): `PalindromicDate.java`

Write a program that prompts the user to enter, *in this order*, (i) a month (which could be in mixed upper-case/lowercase), (ii) a day and (iii) a four-digit year. Then, the program prints the date in a format `MM/DD/YY`, with exactly two digits each for the month, day and year. It also prints whether the reformatted date is a palindrome.

⚠ **Note:** Your program must check for bad input, such as an invalid month name, invalid day of the month, or invalid year (less than 1). In such cases, print only the error message "Bad input." and produce no other output. Your program must handle leap years correctly.

Below are sample runs of the program.

⚠ **Note:** You must use these prompts and output messages exactly as written. The grading system will be looking for this wording.

```
Enter month: FEBruary
Enter day: 18
Enter year: 1979
Reformatted date: 02/18/79
Palindrome? no

Enter month: ocTobEr
Enter day: 11
Enter year: 2001
Reformatted date: 10/11/01
Palindrome? yes

Enter month: APRIL
Enter day: 31
Enter year: 2011
Bad input.
```

Part II: What A View! Wait, What's That Smell? (3 points)

Filename(s): `StonyBrookResort.java`

The Stony Brook Resort and Manure Depot offers a vacation package as follows:

- The regular price for each adult is $125 per day. However, any days beyond the fifth day that an adult stays are 25% off the regular price.
- The price for each child is $75 per day regardless of how many days a group stays.
- A $150 discount is given if the group's stay begins on a Monday and lasts no more than 4 days (including Monday itself).
- A service charge of 5% of the original, non-discounted cost is applied for groups of 6 or more.

Write a program that prompts for, *in this order*, (i) the number of adults, (ii) the number of children, (iii) the length of the vacation in days, and (iv) the day of the week that the vacation begins (which might be given in mixed uppercase/lowercase).

Then, print the following values – again, *in this order* – formatted as dollar amounts with two digits after the decimal point for cents: (i) the total before any discounts have been applied, (ii) discounts (if any), (iii) service charges (if any), and (iv) the grand total.

Here is one test case. You should make up others yourself to fully test your code.

⚠ **Note:** Your program may assume that the user enters only valid input at the prompts.

⚠ **Note:** You must use these prompts and output messages exactly as written. The grading system will be looking for this wording.

```
Enter number of adults: 4
Enter number of children: 3
Enter length of vacation in days: 7
Enter day of week that vacation starts:  Monday
Total before discounts: $5075.00
Discounts: $250.00
Service charges: $253.75
Grand total: $5078.75
```

Part III: Managed Chaos (4 points)

Filename(s): `Payday.java`

A company employs four kinds of employees:

- *managers*, who receive a fixed weekly salary;
- *hourly employees*, who receive a fixed hourly wage for up to the first 35 hours they work and "double overtime" (2x their hourly wage) for overtime hours worked;
- *commission workers*, who receive $250 plus 5.7% of their weekly sales;
- *pieceworkers*, who receive a fixed amount of money per item that they produce (each pieceworker works on only one type of item)

Write a program that computes the net weekly pay for an employee and prints a summary containing, on separate lines *and in this order*, (i) the gross weekly pay, (ii) the income tax withheld, and (iii) the final, net weekly pay (i.e., gross pay minus taxes). The program begins by prompting for the kind of employee, as follows:

```
Choose employee type: (m) manager (h) hourly (c) commission (p) pieceworker:
```

⚠ **Note:** You must use this prompt exactly as written. The grading system will be looking for this wording.

Depending on the kind of employee, it then prompts only for the information required to determine that employee's gross weekly pay. For instance, for a commission worker the program needs to prompt for only the weekly sales (as a `double`) because the $250 and 5.7% amounts are fixed and should be "hard-coded" into your program. Here's an example:

```
Choose employee type: (m) manager (h) hourly (c) commission (p) pieceworker: c
Enter weekly sales: 5420.65
```

After determining the gross weekly pay, the program subtracts 12.5% for taxes and computes the net pay. Assume that any values entered by the user could contain a decimal point. Use `System.out.printf` to format the

three output values (gross pay, taxes and net pay) with a dollar sign and exactly two digits after the decimal point for cents. Below are sample runs of the program.

⚠ **Note:** Your program may assume that the user enters only valid input at the prompts.

⚠ **Note:** You must use these prompts and output messages exactly as written. The grading system will be looking for this wording.

```
Choose employee type: (m) manager (h) hourly (c) commission (p) pieceworker: m ↩
Enter weekly salary: 6700.50 ↩
Gross pay: $6700.50
Taxes: $837.56
Net pay: $5862.94

Choose employee type: (m) manager (h) hourly (c) commission (p) pieceworker: h ↩
Enter hourly wage: 14.25 ↩
Enter hours worked: 56.5 ↩
Gross pay: $1111.50
Taxes: $138.94
Net pay: $972.56

Choose employee type: (m) manager (h) hourly (c) commission (p) pieceworker: c ↩
Enter weekly sales: 5420.65 ↩
Gross pay: $558.98
Taxes: $69.87
Net pay: $489.10

Choose employee type: (m) manager (h) hourly (c) commission (p) pieceworker: p ↩
Enter pieces produced: 156 ↩
Enter pay per piece: 2.50 ↩
Gross pay: $390.00
Taxes: $48.75
Net pay: $341.25
```

Part IV: What Day is It? (4 points)

Filename(s): `DayOfWeek.java`

Write a Java program that asks the user to type in the date and then determines the day of the week. Luckily, there is a clever formula for this purpose

$d = \left(q + \frac{13(m+1)}{5} + k + \frac{k}{4} + \frac{j}{4} + 5j\right) \%7$ (Note that these divisions are all integer divisions.)

- d is 0 through 6 representing Saturday (0) through Friday (6)
- q is the day of the month (1-31)
- m is the month (3: March, 4: April, . . ., 12: December, 13: January, 14: February)
- j is the century (first two digits of year)

- k is the year of the century (last two digits of year)

⚠ **Note:** When the month entered is January or February, we must change the 1 for January to 13 or 2 for February to 14 and then subtract one from the year.

⚠ **Note:** Your program may assume that the user enters only valid input at the prompts. Assume that the minimum year that might be entered is 1600.

A sample run of the program is given below.

⚠ **Note:** You must use these prompts and output messages exactly as written. The grading system will be looking for this wording.

```
Enter year (e.g., 1918): 2015
Enter month (1-12): 1
Enter the day of the month (1-31): 25
The day of the week is Sunday
```

Part V: MyProgrammingLab Exercises & Problems (10 points)

Complete all exercises for Chapters 3 and 4 of `www.myprogramminglab.com`.

How to Submit Your Work for Grading

To submit your `.java` files for grading:

1. Login to `http://blackboard.stonybrook.edu` and locate the course account for CSE 114.
2. Click on "Homework Assignments" in the left-hand menu and find the link for this homework assignment.
3. Click on the link for this homework assignment.
4. Click the "Browse My Computer" button and locate the first `.java` file you wish to submit. **Do not submit `.class` files, zip files, rar files, or any other kinds of file except for `.java` files.**
5. Repeat step 4 for each `.java` file you wish to submit.
6. Click the "Submit" button to submit your work for grading.

Oops, I messed up and I need to resubmit a file!

⚠ When you submit files to Blackboard, the system assembles them into what's called an "attempt". If you need to resubmit your homework, it is essential that you resubmit ALL files again so that Blackboard assembles them as a single attempt. If you fail to do this, the files you uploaded earlier will not be included in the new "attempt" and might not be graded properly.