GEORGE MASON UNIVERSITY | IT 206
Object-Oriented Techniques for IT Problem Solving

Programming Assignment 7/8

Scenario:

Monica is back again from Sew What? Gifts and Knickknacks. She has decided to reward each of her 37 employees with an end of year thank you, week long, family vacation (only one family vacation, up to 10 family members, per employee!). However, since some employees have other plans, not all employees will be able to take Monica up on her offer. In order for her to purchase these vacations in time for the end of year celebration, she needs to have an application created now to track the vacations to be purchased.

Employees must select from one of the following vacation types: Ski Resort or Beach House. All vacations have an employee name, location (e.g. Outer Banks, North Carolina), zip code, and number of family members who will go on the vacation (including the employee!). A valid zip code is only in the format of yyyyy-yyyy, where y is a number.

Beach House vacations have a base cost determined by the number of family members who will go on the vacation:

- 1-3 family members - $196
- 4-7 family members - $306
- 8-10 family members - $524

This vacation also has the option to purchase hurricane insurance. This service ensures a refund of the vacation cost in the event of a hurricane. The cost of this insurance is based on the number of family members who will go on the vacation at a rate of $9.50 for each family member up to 5 family members, and $4.75 for each additional family member above 5.

Ski Resort vacations are held at locations rated 2, 3, or 4 stars. These vacations are charged a flat fee, based on the number of stars of the location, at a rate of $308 per star. For example, a location with 2 stars will have a base cost of $616. A list containing the names of each family member who will go on this vacation, and whether or not they would like to have ski lessons must also be tracked. This will help the Ski Resort determine how many temporary instructors need to be hired, which will be accomplished in a different application. An additional $49.95 is charged for each family member who would like to have ski lessons.

A vacation bill for one vacation should be well-formatted, listing the vacation type (Ski Resort or Beach House) and all information associated to the vacation.

Create an efficient, object-oriented solution, using good design principles and a polished user interface, that will allow Monica (the user) to track all vacations. At the start of the application, the user must be presented with a menu that contains the following options:

1) Add Vacation
2) Remove Vacation
3) Print All Beach House Vacation Bills
4) Print All Ski Resort Vacation Bills
5) Print Vacation Summary
6) Quit

The "Add Vacation" option should allow a user to select the type of vacation they wish to enter and enter in all information associated to that type of vacation. To make each vacation special, Monica has mandated no two vacations can have the same location. Additionally, adding vacations may be entered in any order, but must be stored in lexicographical (alphanumeric) order. If the user already has entered the maximum number of vacations, they should see a message indicating the limit of vacations that may be entered has been reached.

The "Remove Vacation" option should allow a user to see a list of employee names and select an employee whose vacation should be removed. Removing vacations must maintain the lexicographical (alphanumeric) order at all times.

The "Print All Beach House Vacation Bills" option should print only the bills associated to Beach House Vacations.

The "Print All Ski Resort Vacation Bills" option should print only the bills associated to Beach House The

The "Print Vacation Summary" option should create a well-formatted report, listing the total number of vacations, regardless of type, and the total vacation cost across all vacations.

Once a user selects an option other than "Quit", and the option is finished executing, they should be re-prompted to enter another option.

Other Requirements:

- Your solution must use object-oriented techniques, including appropriate constants, constructors, accessors (including a toString() method), validating mutators, and special purpose methods. No points earned for a procedural solution.
- Your solution must contain reasonably **appropriate** validation. Try to think about what might be considered appropriate as you are designing your solution beyond what has already been explicitly provided.
- Your solution must not import any Java library other than JOptionPane.
- Your solution must demonstrate use of arrays.
- Your solution must demonstrate the concepts of abstraction, inheritance, and polymorphism.

- Your solution must be designed as a modular solution using methods other than main, with each method performing one task.
- You may not use Scanner or System classes for input/output. You must use JOptionPane.
- You may not use System.exit, or any variant that exits the program in the middle of the program. The program should only exit once the algorithm has finished completing.
- Your solution may not use any functions or language constructs not covered during IT 106 or this semester's IT 206 without prior authorization from your instructor, even if you know other functions or language constructs. We want everyone to be on the same "playing field", regardless of previous programming exposure, and get practice with algorithmic design to solve problems (the intent of the course). Using something existing not discussed in class does not give you as much practice as solving the problem yourself. **Doing this may lead to a substantial grade penalty, a grade of zero, or an Honor Code inquiry.** When in doubt, ask!

To Do (Check Blackboard for Due Dates):

Programming Assignment 7: Solution Design

1) List and describe the purpose of each class that will be needed to solve this problem
    a. You must separately identify (list) and describe the purpose of each class. One or two sentences per class should be sufficient.
2) *Data Definition Class(es)* - Create a detailed UML class diagram, listing and explaining all class variables, accessors, mutators, special purpose methods, and constructors associated to each data definition class.
3) *Implementation Class* - Create a table that lists all methods that will be used to create the implementation class. For each method identified, provide the following:
    a. A 1-2 sentence describing the purpose of the method
    b. A list of the names, data types, and brief description (1-2 sentences) for each input variable into the method, if there are any
    c. The name and data type of the variable to be returned from the method, or void if nothing will be returned

    Note: You do not need to provide any pseudocode. However, you should create a plan for yourself as to how you will perform the logic for each method. If you do not do this, you will have great difficulty in completing the solution implementation.

    An example format to use for this table is as follows. You can format this table in any way you'd like, so long as the information is clearly presented. Completing this table correctly will help you build your documentation and code for your solution implementation.

Method:	getMagazine
Purpose:	The method will allow for the creation and population of a Magazine object based on user input of a title, cost, and number in stock
Inputs:	none

Return: magazine : Magazine – The Magazine object created and populated from user input

Method: checkout
Purpose: The method will allow for the purchase of a magazine. It takes into account the sales tax that will be charged as part of the cost calculation
Inputs: magazine : Magazine – The magazine to be purchased
salesTax : double – The sales tax percentage to be charged as part of the checkout
Return: void

Upload a Word document containing only items above to Blackboard.

Grading Criteria	
Requirement	**Points**
List and describe the class(es) needed to solve the problem	10
Data Definition Class Design – Detailed UML Diagram	40
Implementation Class Design – Table Listing All Methods	50

Programming Assignment 8: Solution Implementation

Write a well-documented, efficient Java program that implements the solution design you identified. Include appropriate documentation as identified in the documentation expectations document.

To Blackboard, **submit ONLY ONE .zip file** containing all of the .java files part of your submission for your solution implementation. Your .zip file should contain only the .java files in your solution. Be careful that you do not submit .class files instead of .java files.

> **Warning!** You must submit **ONLY ONE** .zip file containing **ONLY** your .java files. Failure to follow this instruction precisely will result in a 10 point deduction of the assignment score. **No exceptions!**
> *Why is this important?* The goal is to teach you how to properly package your IT solutions into a "customer-friendly" format while paying attention to "customer" requirements provided to you.

Your program must compile using jGrasp. Any final program that does not compile, for any reason, will receive an automatic zero. Other IDEs often place in additional code that you are unaware of, doing too much of the work for you. **You are strongly discouraged from using IDEs other than jGrasp.**

Grading Criteria	
Requirement	**Points**
Implementation of object-oriented Java program, using efficient practices, such as the use of constants, good variable names, information hiding, no redundant code, etc.	70
Appropriate objective-style documentation	10
Appropriate intermediate comments	20