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

Programming Assignment 1/2

Scenario:

You are working for GMU Retirement Planners, Inc., a student-run company that helps customers plan retirement parties. The company has to track all retirement parties for which it is providing planning services. However, since you are building a brand new application for the company, you decide to first build a proof of concept application that only manages one retirement party for now, with the goal of tracking all retirement parties in the future. Each retirement party has a customer budget that describes the maximum amount of money the customer is willing to spend on items purchased for the retirement party. The application must track a running total of the amount of money spent on items purchased for the retirement party and the number of items purchased.

Create an application that uses solid object-oriented principles to simulate one retirement party and an accompanying implementation class that will instantiate one instance of the data definition class, using it to process item purchases for a single retirement party.

The data definition class must also store the name of the person retiring and if a speech will be given. It must have validating mutators for all numeric instance variables and non-validating mutators for non-numeric instance variables. Accessors must be created for all instance variables. The data definition class must allow for the budget to only be set at object creation time, but should not be required. The default budget when it is not specified at object creation time should be $1,000.00. Additionally, the data definition class must be able to calculate how much money is remaining in the budget at any given time.

The implementation class will instantiate one instance of the data definition class with a budget of $2000.00. The program will then prompt the user for the name of the person retiring and if a speech will be given. Next, the user will be prompted to enter item purchase amounts (one at a time) until they have indicated they are finished entering item purchase amounts. Validation of each purchase amount must include a check to make sure there is sufficient money remaining in the budget. When the user has indicated they are finished entering purchase amounts, the program will display a well-formatted summary of the name of the person retiring, whether a speech will be given, the total number of item purchases, the amount of money spent on all item purchases, and the amount of money remaining in the budget.

Notes:

- There are a number of validations that must occur. As one example, think about what type of validation might be appropriate when gathering an item amount. Make sure these are all handled. Don't forget about what you learned about data validation in IT 106 (e.g. if statements, try/catch, etc.). Whenever an invalid value is entered, the user must be informed that an error has occurred and then be re-prompted to enter in a new value.
- Think about any special purpose methods that might be needed in the data definition class. Special purpose methods are important when using values in instance variables to perform a specific process.
- Think about how constructors might be used when creating your data definition class.

Other Requirements:

- Your solution must use object-oriented techniques. No points will be awarded for a procedural solution.
- 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 1: 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) 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) Create a defining diagram/IPO hierarchy chart detailing the implementation design (**Note:** You must do this, but it is not required to be submitted)
4) Create pseudocode detailing your solution design
    a. Data Definition Class(es) – You must provide pseudocode ONLY for validating mutators and special purpose methods. No pseudocode is necessary for constructors or accessors
    b. Implementation Class – You must provide pseudocode for **all** methods

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
Detailed UML Diagram	40
Pseudocode	50

Full points will be awarded for an accurate, efficient, complete solution design. Partial credit is available.

Programming Assignment 2: 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.

Full points will be awarded for an accurate, efficient, complete Java program **that compiles using jGrasp**. Partial credit is available. 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