**CMSC203**

**Assignment #2**

**Birthday Gifts**

**Assignment Description**

You have decided to use a specialized toy company to purchase birthday gifts for the young children of your friends and relatives. The toys you can choose are divided into three categories: **plushies**, **blocks**, and **books**. You can add a card and/or a balloon with each gift. Your program must print a title and a list of all the gifts to the console, with a total for each gift, and the total amount of the order. Your program must generate a random five-digit number to serve as the order number, followed by your name as programmer. The **Toy** class provided, will keep a total cost for one toy and determine if the toy is age-appropriate for the child. You should **not make any changes to the Toy class**. Assume that the user enters the name and age of the child correctly.

**Concepts tested by this assignment**

* Input validation loops
* Relational and logical operators
* Use a worker class
* Create a driver class
* Currency format
* Random number generation
* GitHub

**Classes**

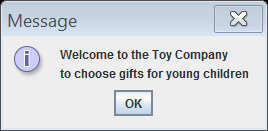
Create a driver class called **Birthday**. This class will use the **Toy** class provided for you. Refer to Assignment Details for implementing the **Birthday** class.

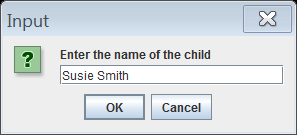
You should **not make any changes to the Toy class**.

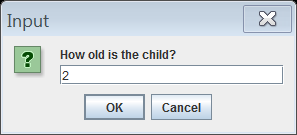
**Assignment Details**

* All input items and messages will use Dialog boxes. Only the title, list of gifts, total, order number, and programmer name will be printed on the console. NOTE: if you use a Mac, you must use Alerts and TextInputDialog; if you use a Windows machine, JOptionPane is slightly simpler, or you could use the Alerts/TextInputDialog option. (See Week 2, Module 4)
* Use a Dialog box to display the welcome message.
* Ask for the name of the child.
* Ask for the age of the child.
* Ask for the toy choice and validate the input choice.
* Print out a message if the toy is not age appropriate and ask if the user wants to cancel that toy request.
* If the user replies "yes", repeat the steps for the name, age, and toy choice. If the user replies "no", process the toy requested.
* Ask if a card or balloon should be added to the gift.
* Display on the console the name, age and the total for gift.
* Ask if another gift is desired. If yes, repeat the steps starting with the name of the child.
* If no, display on the console the total amount of the order, a random five-digit order number, and the programmer name.
* All costs displayed on the console should use currency format.
* Write about your Learning Experience, highlighting your lessons learned and learning experience from working on this project. What have you learned? What did you struggle with? What would you do differently on your next project? What parts of this assignment were you successful with, and what parts (if any) were you not successful with.
* GitHub: In your repository (see Lab 1), upload the files initially provided in Blackboard for the project. When you are finished with the design and programming, upload your design file and java files. You will want to upload these files as contents of a directory so that future uploads can be kept separate.

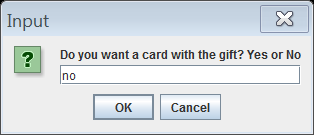
**Examples – Tests**

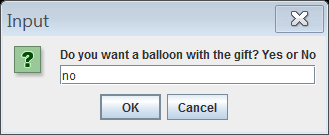


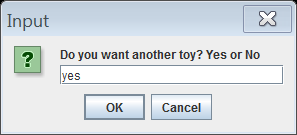












**Sample console output:**

BIRTHDAY GIFTS

The gift for Susie Smith 2 years old is plushie $25.00 The gift for Johnny Jones 5 years old is blocks $20.00

The total cost of your order is $45.00 Order number is 19578

Programmer: insert your name here

**Good Faith Attempt**

To satisfy the “Good Faith Attempt” (see Blackboard) your code must compile and print the output without finding the name, age, type of toy, cost, or order number. Your GFA need not show the prompts, create the Toy object, or compute the cost or order number. The output for the GFA will be:

The gift for x 0 years old is blocks $0.00 The total cost of your order is $0.00 Order number is 0

Programmer: insert your name here

Note that the GFA is not graded, so you must submit your totally-working code by the submission deadline to have a non-zero grade.

**Test Plan**

Test your program with at least 3 test cases. **Make sure your tests cover all the possible scenarios.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #** | **Input** | **Actual Input** | **Expected Output** | **Actual Output** | **Did the test pass?** |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

**Deliverables**

Design: Submit your initial design (pseudo-code and test cases) in Blackboard.

Implementation:

* Turn in your final design (pseudo-code and test cases)
* Turn in the working java file Birthday.java (not the .class file).
* Learning Experience: highlight your lessons learned and learning experience from working on this project. What have you learned? What did you struggle with? What will you do differently on your next project? Include what parts of the project you were successful at, and what parts (if any) you were not successful at.
* GitHub: In your repository (see Lab 1), upload the files initially provided in Blackboard for the project. When you are finished with the design and programming, upload your design file and java file. You will want to upload these files as contents of a directory so that future uploads can be kept separate. Take a screen shot of the GitHub repository.

**Submission Detail**

* Upload two .zip files for each assignment. The first .zip file will contain **all** the files required for the assignment (final design with pseudocode and table of test cases, screen shot of runs and the GitHub repository, and the .java file, and will be named: LastNameFirstName\_Assignment2.zip.
* The second .zip file will only contain the .java files and will be named: LastNameFirstName\_Assignment2\_Moss.zip. **This .zip will not have any folders in it – only .java files.**