Queens College, CUNY, Department of Computer Science

Software Engineering

CSCI 370

Summer 2019

Instructor: Dr. Sateesh Mane

Demo of GUI in class, Monday 7/15/2019 – Thursday 7/18/2019 (see below)

Zip archive due by 11:59 pm on the day before your in-class demo

1 Project 1

- **All students work individually for this project, there are no teams.**
- *This project does not require a database or other persistent storage medium.*
- **Code must be written in Java and must work when loaded into Eclipse.**
- Please submit your solution via email, as a file attachment, to `Sateesh.Mane@qc.cuny.edu`. The file attachment is a zip archive with the following naming format:

  `StudentId_first_last_CS370_project1_Spring2019.zip`

- **Important: Do not encrypt the zip. Other formats such as RAR archive or OneDrive or Google Drive, etc. will not be accepted.**
- **Your project zip archive must contain all your program source code.**

  1. Include all the required '.java' source files.
  2. Do not submit compiled '.class' files.
  3. **I must be able to load your source files into Eclipse and run your GUI.**
  4. Also include screen shots of various stages of a purchase of an item (see Sec. 1.2 below).
  5. *Low resolution images (small file size) are acceptable.*
  6. Include a cover document ('cover.pdf' or 'cover.docx') describing your work. Explain your software architecture and your unit testing/debugging. *You may include your screen shots as images in the cover document.*
  7. **Bonus: include a '.jar' executable file which I can double-click and run directly.**

1.1 Grading policy

- The lectures during the week Monday 7/15/2019 through Thursday 7/18/2019 will be reserved for student demos. During one of the above lectures, you must present a demo in class, of your implementation of the project.

  1. I will ask you questions and/or run my own tests of your GUI in class. *You must be able to answer all questions pertaining to your submission.* **Students who present a demo but cannot explain their work to me personally in class will receive an automatic failing grade.**
  2. On each day Monday–Thursday 7/15/2019–7/18/2019, at the start of the lecture I shall take a head count and the names of all the students who wish to demo their project that day. The time available for each student demo will be equally divided between all students presenting on that day. If too many demos are presented on the last day (for example), there may be little time for each demo, which could result in demos missing grade points for parts that were not presented/tested.
  3. The date on which you present your demo will be recorded. A student who presents a demo on an earlier day may receive a higher grade than a student who presents a demo of equal quality but on an later day.
  4. Your zip archive must be emailed to me not later than 11:59 pm on the day before you present your demo in class. (That is to say, you are not permitted to make changes to your project after you demo it in class.) **Students who submit their zip archives after presenting their demo in class will receive a grade cap of C, regardless of the quality of their solution.**
  5. **You may present your demo only once.** You are not permitted to present a repeat demo with bug fixes or enhancements, etc. *Your project submission is expected to be complete and correct at the time of your demo in class.*
  6. **Your grade will be determined principally by your presentation in class.** (Subject to caveats about your zip archive and "interference" as explained below.) I will use your zip archive to confirm your demo in class and to perform additional tests.
  7. **Students who do not present a demo in class will receive a failing grade, regardless of the quality of the work submitted in their zip archive. It is not acceptable to submit only a zip archive and not present a demo in class.**

- **Any attempt to interfere with the presentation of student demos will be treated as a serious offense and will incur a serious grade penalty, including a failing grade.** In this context, "interfere" includes the following:

  1. Attempts to obtain more than your allotted time to present your demo, thereby denying other students their fair share of time.
  2. Making comments on the work by other students and/or interrupting conversations between myself and other students when I examine the work of other students.
  3. "Cutting the line" if I call upon another student to present their demo ahead of you.
  4. Creating a disturbance in the class, which obstructs my ability to examine all student demos fairly.

1.2 Project specification

- This project is to create a GUI for a vending machine, with a user interface.
- **Mandatory: everything should be displayed in one GUI.**
- The GUI displays a list of items for purchase and the price of each item.
  1. The items may be in a list, else a two dimensional array.
  2. Each item should have a descriptive name (avoid unnecessarily long/complicated names).
  3. The price of every item should be a multiple of 10 cents.
  4. No item should cost more that $5.
- The user selects one item: the details how to implement the selection are up to you.
  1. One possibility is that each item has a button, click on the button.
  2. Another possibility is that each item is indexed with a code, for example "A1" or "E9" etc., and the user enters the code in a keypad on the side of the GUI.
  3. **Only one item may be selected at a time.**
     Implementations which allow selection of multiple items will incur a loss of grade.
  4. Assume the machine never runs "out of stock" so all items are always available to buy.
- **The GUI must offer at least eight items, with a selection of different items.**
  1. At least one drink: soda or soft drink, energy drinks, also mineral water.
  2. At least one chocolate item (or candy bar).
  3. At least one "bag of nuts" item, peanuts/almonds/granola bar, salted/unsalted, etc.
  4. At least one "bag of chips" item, potato chips, cheetos, doritos, etc.
- **Note: Describe each item with simple (brief) text wording, e.g. "Pepsi" or "salted peanuts" etc. Do not display images or icons, etc.**
- **You may include other types of items but note the following.**
  1. **All items must be restricted to what real vending machins offer: no fresh fruits or vegetables, meat, etc.**
  2. **No alcohol or cigarettes/cigars or medicines or other restricted items.**
  3. **No lottery tickets, or gambling in general.**
- After a selection has been made, the selected item should be highlighted in the GUI.
  1. The details of how to implement "highlighting" are up to you, but it should be easily understandable to the user.
  2. The user is permitted to change the selection, i.e. which item he/she wishes to buy.
  3. **Hence the GUI must have a button ("Done" or "Buy" etc.) to indicate the user has made a final decision to purchase the selected item.**

- After the user has clicked the "Buy" etc. button, the GUI should display a message instructing the user to pay the relevant amount of money.

    1. The GUI should have an interface for the user to make payment.
    2. Payment is by cash only, with a display ('nickel', 'dime', 'qtr', '$1', '$5'), there are no pennies and no 50-cent coins, also no bills higher than $5.
    3. The user inputs the number of nickels, dimes, quarters, dollar bills and $5 bills he/she wants to enter and clicks a "Pay" or "Enter" button.
    4. The vending machine (i.e. your software) adds up the total amount of money entered.
    5. The machine calculates the change and (i) displays a message that the item has been delivered and (ii) the amount of change returned.
    6. The amount of change returned can be displayed as a decimal number.

- **There must be a "Cancel" button in case the user decides not to buy the item.**
- **After a purchase has been completed or canceled, the GUI returns to its initial state.**
- The machine has no memory of previous purchases, etc.
- *This project does not require a database. No information is saved between start/restart, etc.*
- **Bonus: Display the change returned as (number of dollar bills, quarters, dimes and nickels).**
- General comments:

    1. The GUI should have the appearance of a real vending machine: upright/rectangular.
    2. However, do not waste time on overly fancy colors or borders, etc.
       Credit will be awarded for functionality and user friendliness, not for fancy colors and borders.
    3. The overall GUI display should look similar to the following:

<table>
<tr><td rowspan="2">Items Display</td><td>(messages)

Payment Section</td></tr>
<tr><td>(message: "Delivered")</td><td>Change</td></tr>
</table>

1.3 Notes

- Some students experimented with the vending machine outside the lecture room. They selected an item which cost $2 and paid only $1 and tried to cancel the deal. The machine swallowed the $1 and refused to return it. Once you pay money you must go through with the purchase.
  We shall stick with the policy that if the user hits "Cancel" then the user gets his/her money back.