[Please read all 3 pages of this document first]

COSC1519 Introduction To Programming

Assignment 1: HomeAutoBot (10 marks)

[spec version 2020.02.29; ensure you have the latest version]

Due dates and times: See section 3 of this document

Note: You are awarded marks based on your ability to follow and fulfill the given requirements of this assignment. **Every word in this document matters.** If it is not written, it is not a requirement but you must use the necessary concepts shown in class materials as the list of things you must not do is too long to list. If there are changes to any requirements or additional clarifications, announcements will be made via *Canvas→Announcements*.

Plagiarism warning: This is an individual assignment; all submitted work must be your own. Do not ask others to write any part of your code. **Do not even let others see your code.** The university takes plagiarism and intellectual property matters very seriously; plagiarism attracts academic penalties (e.g. zero marks or more severe penalties) and disciplinary warnings that go on your record. See RMIT Academic Integrity information for specific details.

Tip: Books and lecture notes do not contain solutions to programming tasks. Instead, programmers learn concepts and apply them to solve unforeseen programming tasks. In this assignment, you will be demonstrating your understanding of the concepts covered in first few week of Introduction To Programming lectures. There is no need to wait to get started as you can attempt some parts already. Students are expected to start on released assignments and are expected to manage their time and minimise risks that might prevent timely completion of the work. As this is tertiary education, you are also expected to independently investigate additional examples and perform the necessary research to complete this assignment. Hints and tips will be given during classes and the class work will help you complete this assignment. You are expected to do the class work and the assignment, simultaneously.

1. Introduction

You must create a prototype of a simple home automation bot named “Fraizr” (fictional name) that demonstrates the exact behaviours shown during the week 1 lecture demonstration of the assignment.

The code that you write does not need to control any appliances. Fraizr has a list of appliances that it can either turn on or turn off (see video above).

You are given marks on your ability to follow all requirements on this document as well as on your ability to match behaviours in the said demonstration. When in doubt, please ask via the Canvas→Discussions→Assignment 1 discussion forum.

Note: When clients explain their requirements, it is normal for the programmer to refer to the presentation several times and ask questions as needed.

2. Functional and Development Requirements

All inputs and outputs of your program should be via `GTerm`. For this, make a copy of the Week01/Tutorial01 Eclipse project from Canvas→Assignments→Tutorial 1. If you have created a dummy project named 'Assignment 1' as a part of Tutorial 1's instructions, delete it or rename it to something else. Rename the copy of Week01/Tutorial01 project and name it "Assignment 1". Now rename the .java file under 'src' to HomeAutoBot.java. This is the file that you will submit.

In addition to following good coding practices taught during lectures, Fraizr must be developed in 3 stages discussed further below; Each stage will guide you correctly to the next stage so it is crucial that you follow this sequence. However, only the final stage that you complete must be included in the HomeAutoBot.java file.

All 3 stages follow the following algorithm (identify the steps from the lecture 1 video):

A. Show welcome message.
B. Input appliances
C. Listen for instructions mode until user chooses to exit by selecting the cancel button.
    1. Display appliances and their current states (exactly as shown in video) and ask for input
    2.1 If user input starts with "turn on" or "turn off" followed by an appliance name, display confirmation and update state variable/array element.
    2.2 If user input starts with "turn on" or "turn off" followed by an unknown appliance name, display "I can only control: ..." followed by all appliances and their states.
    2.3 If the user input does not start with "turn on" or "turn off", display "please say turn on or off only" and display all appliances and their states.
D. Show goodbye message.

Development Stage 1: Fixed to 4 appliances. Appliance names are entered by the user. Does not use arrays; variables must be created for appliance names and appliance states (on/off). Comments are added to the code explaining where each algorithm step is located in the code.

Development Stage 2: Fixed to 4 appliances. Appliance names are entered by the user. Must use 2 arrays: one for appliance names, one for appliance states. Comments are added to the code explaining where each algorithm step is located in the code then discusses coding approach.

Development Stage 3: At the start, asks user the number of appliances and then takes name inputs for that many appliances. Must use 2 arrays (see stage 2) and each array's length matches the number of appliances entered. Comments are added to the code explaining where each algorithm step is located in the code then discusses coding approach and evalauates against alternatives.

Colors: For all stages, you must use exactly 5 different font colours to differentiate between the following types of text:

1. Mode headings (e.g. greeting, input taking mode, listening mode, goodbye message)
2. Appliance names.
3. The word "on" when state of an appliance is shown.
4. The word "off" when state of an appliance is shown.
5. All other text shown inside GTerm.

You must choose your colours so that it is easy for the end-user to read. It should be possible to change colours of all text of a certain type from one place (Tip: Variables!).

3. Delivery/Submission Requirements

You to perform two deliveries of your work. There are **three important dates/actions** to remember.

- *Delivery 1*: Submit a partially complete HomeAutoBot.java via Canvas→Assignments→Assignment 1 (not week 4) just before your allocated practical class and demo it during your allocated practical **in week 4**. Completion of *Delivery 1* attracts **3.75 marks**. Your demo instructor will ask you questions related to your work.
- *Delivery 2*: Submit only the HomeAutoBot.java of the highest working stage (and image files if doing the bonus parts) **by end of week 5** (before Monday of week 6), via Canvas→Assignments→Assignment 1 (not week 6) then demo the work that you have done after delivery 1, during your allocated practical **in week 6**. The obtainable marks for *Delivery 2* is as follows:
  - **6.25 marks** if HomeAutoBot.java stage 3 requirements completed in final submission and demo'ed in week 6.
    or
  - **3 marks** if only up to and including stage 2 requirements completed, submitted in HomeAutoBot.java via Canvas and demo'ed in week 6.
    or
  - **1.5 marks** if only up to and including stage 1 requirements completed, submitted in HomeAutoBot.java via Canvas and demo'ed in week 6.

Deductions:

- -2.5 marks: If code does not use GTerm for all inputs and outputs.
- -1 mark: If code does not use colours as required.
- -1 mark: If code uses unsuitable types of loops or inefficient code.
- -1 mark: Bad variable names, lack of comments near code blocks and other bad practices.

Bonus marks:

To receive any bonus marks, you need to have received full marks for stage 3 and all non-bonus components of this assignment.

+1 bonus mark: Display images for appliances. User chooses the image files. Inputs of images and displaying of images must be done only using GTerm methods. If using other's images, you must give a full reference in the comments of your code.

+1 bonus mark: Keep track and display the total running time of each appliance whenever the appliance name is shown (next to the appliance's current state). Timer for each appliance should start only when it is turned on and must stop immediately when it is turned off. If the appliance is turned on again, the timer continues from where it stopped.

Important note: If there is no submission before the start of your class, you will not get a turn to demo. If there is no demo, there will be no mark for that entire delivery. You will need to bring your own laptop to the demo. Code with any invalid Java (e.g. red dots in Eclipse) will receive 0. Code that crashes at run-time will receive a 50% penalty of the eligible mark of that delivery.

Late submissions: Late submissions incur a standard 10% penalty for each working day late for that delivery; submissions will not be accepted after 5 working days from deadline. If you wish to organise special consideration, please contact RMIT special consideration.

End of Document