Curtin College in association with Curtin University

FUNDAMENTALS OF PROGRAMMING ASSIGNMENT TRIMESTER-Three 2020

DUE: Sunday January 3, 23:59 UTC+8

BACKGROUND

In your practicals, you have learned about how to use programming to model problems and simulate real world behaviours and soon you will learn about automation. In this assignment, you are to develop a model and then utilise it in simulating a solution to the given problem.

THE PROBLEM

COVID-19 Vaccination

Corona Virus Pandemic situation is not hidden from anyone in the world. The situation is coming into control after the release of Covid-19 Vaccine. Let us assume that the Airport is allowing the passengers who are Covid-19 positive after providing the Vaccination before travel to be able to safeguard the passengers who are travelling to national and International destinations. The Vaccination is limited and should be only given to the patients who are positive. To be able to check if the patients are positive you will have to check the following conditions:

The main things to be checked are high fever and other symptoms (dry cough, sour throat and respiratory problems). In this assignment we will design a program to check the symptoms of people entering the airport and hence decide if they need Vaccination before travel or not.

In addition to the above mentioned problem we have Airport Conveyer belts problem. Airport Conveyer belts are an integral part of the luggage management system on any Airport. The identification of over-sized and undersized luggage is mostly done manually and not fully automated. We will simulate and automate the process of airport conveyer belt in this assignment.

SPECIFICATION

The assignment is divided into two tasks. Note that task 1 needs to be done first followed by task 2. You have got an input file “Assignment_data_input.csv” attached to this assignment specification which you will use for both of the tasks. The input file has the data arranged in the following format:

First Name	Luggage weight	Luggage dimensions	Temperature readings in Celcius	Dry Cough	Sour Throat	Respiratory problems

TASK 1: Corona Virus Vaccination at the Airport (30%)

The attached input data file "Assignment_data_input.csv" contains the information about the passenger's temperature readings and other Covid-19 symptoms. Using that data identify the passengers who should get Vaccination and who should not. Only the passengers with positive Covid-19 will get the vaccination. The first check should be done for the temperature and if the temperature is above the normal temperature then the other symptoms should be checked. If the temperature is normal then we do not need to check for other symptoms and passengers should be allowed to travel without the Vaccine. The normal temperature should be in the range of 36 Celsius to 37 Celsius. Any temperature above this range should be considered doubtful and hence other symptoms should be checked. Once the temperature check is done then the other three symptoms (Dry Cough, Sour throat and respiratory problems) should be checked. For the conformation of Corona virus, the two symptoms should be yes from the three symptoms mentioned above otherwise the passenger is safe to travel and no vaccination is required. After classifying the passengers the affected passenger's data should be put in a different output file so that the Vaccination is provided to them. (We will assume that the patients are recovered instantly after the vaccination and allowed to travel- this is a hypothetical situation)

For full marks on this task, you need to output the correct number of passengers who have Covid-19 symptoms and need Vaccination in the **Vaccine.csv file**.

Note: You can do the coding using Vim on terminal or Jupyter notebook.
Note: You can use procedural programming or Object Orientation programming for this task (both are acceptable). However for the next task it is mandatory to use the OO programming.

TASK 2: Airport Conveyer Belt Simulation (30%)

Airport Conveyer Belt: After the Vaccination process is over, all the passengers are allowed to travel. The attached input file contains the data about the luggage of passengers. Depending upon the weight and dimensions of the luggage, it should be classified into under-weight, appropriate weight and over-weight luggage. You need to check for the weight and dimensions for the luggage. The luggage is divided into four categories in the attached "Assignment_date_input.csv" file. The categories are Very Small, Small, Normal and Big. For the ease of programming the actual dimensions have been avoided. After the identification of luggage it should go to the appropriate belts as described below:

a. C1 (Conveyer belt for under-weight luggage)

(Assumption: We assume that the under-weight category is less than 30 Kgs and the dimensions of the allowed luggage can be small and normal)

b. C2 (Conveyer belt for appropriate-weight luggage)

(Assumption: We assume that the appropriate -weight category is equal to 30 Kgs and the dimensions of the allowed luggage can be small and normal)

c. C3 (Conveyer belt for over-weight luggage)

(Assumption: We assume that the over -weight category is greater than 30 Kgs and the dimensions of the allowed luggage can be normal and big)

d. Identified as CL (Cabin Luggage)

(Assumption: We assume that the Cabin Luggage category is less than equal to 7 Kgs and the dimensions of the allowed luggage can be very small only)

You will have to figure out where should the luggage go and then display where the luggage finally is. You should have 4 output files for the 4 categories mentioned above. Design a menu from where we can go to any of the output file and see the data (which luggage went to which belt along with the other related data of the passenger).

For full marks in this section you should have the correct number of luggage items in each output file. The four output files of the task 2 should be in .csv format.

Note: You can do the coding using Vim on terminal or Jupyter notebook. For this task you should use classes and objects (eg: passenger and luggage objects) and Object oriented programming concepts.

……

You can have two different code files for two different tasks.

There should be 5 output files from two tasks mentioned above- one from task1 and 4 from task 2.

USER GUIDE AND REPORT (35%)

You need to submit your User Documentation and Report in doc/docx, pdf or Jupyter Notebook format.

Your **User Documentation** will be minimum 3-5 pages, with a section for each simulation and should include the following:

- An **overview** of each of your program's purposes and features.
- A **guide** on how to use your programs/simulations.
- A **discussion** of your code, explaining how it works, any additional features and how you implemented them.

The **Corona Virus Vaccination** and **Aircraft conveyer belt report** will comprise of a mini-paper that is 2-3 pages long and follow the structure of a standard academic report. Required sections are:

- **Abstract:** Explain the purpose of the simulation and state the parameters you have considered in designing the simulation, and the outcomes/recommendations.
- **Background:** Discuss the purpose of the program/simulation and your choice of parameters.
- **Methodology:** Describe how you have chosen to set up the simulation, and why. Include commands, input files, and outputs – anything needed to reproduce your results.

- **Results:** Present the results of your simulations – include tables, plots (optional) and discussion.
- **Conclusion and Future Work:** Give conclusions and what further investigations could follow.
- **References**

(You can find examples on google but do not copy and paste. Just use the examples as reference)

CODING STANDARDS (5%)

Your code submission must conform to coding standards emphasised in the lecture and practicals. Remember: consistency is key!

SUBMISSION

Submit electronically via Moodle. Make sure to submit early. You can submit multiple times – we will only mark the last attempt. Take care not to submit your last version late though. Read the submission instructions very carefully.

You should submit a single file, which should be zipped (.zip). The file must be named FOP_Assignment_<id> where the <id> is replaced by your student id ignoring the angle brackets. There should be no spaces in the file name; use underscores as shown.

The file must contain the following:

- Your **code**. This means all files needed to run your program. That includes input files used as part of the assignment if that is required to run your program.
- **README** file including short descriptions of all files and dependencies, and information on how to run the programs.
- **User Documentation**, as described above.
- A signed and dated **cover sheet**. These are available on Moodle You can sign a hard copy and scan it in or you can fill in a soft copy and digitally sign it.

Make sure that your zip file contains what is required. Anything not included in your submission may not be marked, even if you attempt to provide it later. It is your responsibility to make sure that your submission is complete and correct.

REQUIREMENTS FOR PASSING THE UNIT

Please note: As specified in the unit outline, it is necessary to have attempted the assignment reasonably in order to pass the unit- **Assignment is a Unit Hurdle**. Section 2.5 of Unit outline explains the weightage associated with the assignment which is 30% of overall marks. As a guide you should get 45% in the assignment to pass this Unit. Note that the marks indicated in this section represent maximums, achieved only if you completely satisfy the requirements of the relevant section.

Plagiarism is a serious offence. This assignment has many correct solutions so plagiarism will be easy for us to detect (and we will). For information about plagiarism, please refer to http://academicintegrity.curtin.edu.au.

In the case of doubt, you may be asked to explain your code and the reason for choices that you have made as part of coding to the unit coordinator. A failure to adequately display knowledge required to have produced the code will most likely result in being formally accused of cheating.

Finally, be sure to secure your code. If someone else gets access to your code for any reason (including because you left it on a lab machine, lost a USB drive containing the code or put it on a public repository) you will be held partially responsible for any plagiarism that results.

LATE SUBMISSION

As specified in the unit outline, you must submit the assignment on the due date. Acceptance of late submissions is not automatic and will require supporting documentation proving that the late submission was due to unexpected factors outside your control. See the unit outline for details as to the procedure for requesting that an assessment be accepted after the due date.

Also note that IT related issues are almost never a valid excuse.

In the event that you submit your assignment late and are deemed to have a valid excuse, you will be penalised 10% (that is, 10% out of 100%, not out of what you would have received) per calendar day that you are late, up to a maximum of seven (7) calendar days. Any work submitted after this time will not be marked and you will automatically fail the unit. Note that if you are granted an extension you will be able to submit your work up to the extended time without penalty – this is different from submitting late.

CLARIFICATIONS AND AMENDMENTS

This assignment specification may be clarified and/or amended at any time. Such clarifications and amendments will be announced on the unit's Blackboard page. These clarifications and amendments form part of the assignment specification and may include things that affect mark allocations or specific tasks. It is your responsibility to be aware of these by monitoring the Assignment Assessment Blackboard page.