Curtin College Singapore
FOP1005 – Fundamentals of Programming

Assignment 1

30% of the unit Grading

Time Frame:

Release Date: 17th Mar 2023, 10:00 am (UTC+8)

Submission Date: 07th Apr 2023 10:00 pm (UTC +8)

Contents and Percentage Distribution:

The details of important components of assignment and their constriction for the marking of this assignment are as below.

Sr.	Content	Percentage Distribution (if any)	Page No
1.	Basic Details	NA	1
2.	Overview	NA	2
3.	Specifications	NA	2
4.	Reading CSV File	10 %	2
5.	Displaying Menu	04%	2
6.	Adding Filters	20%	3
7.	Printing Names	05%	3
8.	Graphing Results	20%	3
9	Reset Filters	01%	4
10.	Coding Standards	10%	4
11.	User Documentation and Report	30%	4
12.	Submission Details	NA	5
13.	Submission Instructions	NA	5
14	Unit Passing Requirements	NA	5
15.	Late Submission	NA	6
16.	Clarification and Amendments	NA	6
17	What should be Avoided	NA	6

*All time details mentioned are in UCT+8 time zone standard
*Read the complete document and submission requirements before start working on it.
*This is an open book open computer/internet assessment however, any direct code copying form unacceptable links/repositories will be subjected as an academic integrity issue.

1 Overview

You are a freelance Data Analyst. You are required to analyse the dataset of video games sale and generate a detailed report on popularity and in market rating of the game. More details on the dataset provided can be found here: https://www.kaggle.com /datasets/gregorut/videogamesales. You should use the dataset provided on the unit Moodle page. The assessment has two parts as (a) coding part and (b) documentation and report. All parts have marks assigned an should be completed as per description provided for each part.

Note: use the cleaned data provide on the unit Moodle page

2 Specifications

The coding portion of this assignment is broken up into four main tasks. There is also a report.

section of the assignment. The four main coding tasks are:

1. Reading games sale data stored in the GamesSales.csv file.
2. Displaying a menu asking the user what filters user want to apply to the data.
3. Displaying the data matched by the set filters as a graph or by printing to the terminal.
4. Reset filter by removing all the filters that has been applied.

2.1 Reading CSV File (10%)

The GamesSales.csv file contains all the details of the games and sales considered by the company as a comma-separated value (CSV) file. Before going on to the next tasks you'll need to read this file and store this data in a list or an array. Discuss which option you chose and why in the User Documentation.

There are 9 attributes that describe a given game shown in the GamesSales.csv file, they are:

- Name: In market name of the game.
- Platform: The gaming platform for the game is designed (e.g., x360 PS5)
- Year: Year of release.
- Genre: Category based on recent games classification. (e.g., Sports)
- Publisher: Agency responsible for publication of the game.
- NA_Sales: Sales in North America (in millions).
- EU_Sales:. Sales in Europe (in millions).
- JP_Sales: Sales in Japan (in millions).
- Other_Sales: Sales in the rest of the world (in millions)
- Global_Score: Total worldwide sales.
- Critic_Score: Critic score given to each game.
- Critic_Count: Number critic contributed for the rating.
- User_Score: User rating for each game.
- User_Count: Number of users contributed to the rating.
- Developer: The name of developer.
- Raring: Overall market rating.

2.2 Displaying a Menu (04%)

Once you have read in the data from the file, your next task is to display a menu asking the user which of the following options they wish to perform:

1. Add filters: User select a smaller portion (based on filters) of data to print or graph.
2. Print names: Prints the names and descriptions (attributes) of the filtered games (based on the filter).
3. Graph data: Graphs the filtered data (based on the filters).
4. Reset filters: Removes all the applied filters.
5. Exit: Exits the program.

2.3 Adding Filters (20%)

This menu option allows the user to add filters to the data read in from the CSV file to help gain insights into the data. The user should have the ability to add filters on **Platform, Genre, Year (range) Global_Score, and Rating**. The user should be able to add filters on multiple of these attributes. This is how adding a filter to each attribute should work:

- **Year (range):** The user is shown the range (minimum year and maximum year) of years the data is analysed and select one. Once the year is selected then the data selected should be used for any further filters' application, printing/showing graph. All filters should be applied as a code (should not be hardcoded). In other words, if the data is changed (added a new attributes) in the file then program should display the updated details without you needing to change the code.

- **Platform:** The user can choose a platform, only the game with platform (selected) should be used for any further filters or printing or showing graph. Different games should be selected from the file (based on data present), should not be hard coded.

- **Global_Score:** The user can choose a Global_Score, only the games with Global_Score (selected) should be used for any further filters or printing or showing graph. Different games should be selected from the file (based on data present), should not be hardcoded.

- **Genre:** The user can choose a Genre, only the games with Genre (selected) should be used for any further filters or printing or showing graph. Different games should be selected from the file (based on data present), should not be hardcoded.

- **Rating (range):** The user is shown the range (minimum value and maximum value) of rating the data is analysed and select one. Once the rating is selected then the data selected should be used for any further filters' application, printing/showing graph. All filters should be applied as a code (should not be hardcoded). In other words, if the data is changed (added a new attributes) in the file then program should display the updated details without you needing to change the code.

•

2.4 Printing Names (05%)

This option should print names and descriptions (name, platform, year, etc) of the games remaining after the filters are set. No other information needs to be printed other than the name followed by the associated description.

2.5 Graphing the Results (20%)

This option should graph the data remaining after the filters are set. When the user chooses this option they should be asked what data they want to plot on the x-axis. The options are platform, year and genre. The y axis should always be the Global_Score and Rating for multiple sales details for a game (by platform, year and genre) only highest should be plotted/printed. The x-axis and y-axis should be labelled, and the graph should have a title. It is up to you as to 4 what type of graph you want to use to plot the data (bar, line, histogram, etc) but you should use at least 2 different types. Bear in mind that some graph types are better suited for different types of data. Explain in your User Documentation what graph types you used and why.

2.6 Reset filters (01%)

This option should remove all the filters that has been applied so users can now select new filters without quitting your program.

2.7 Coding Standard (10%)

Your code submission must conform to coding standards emphasised in the lectures and practicals. Your code should also make use of multiple functions to reduce code duplication. Your code should also follow the program layout structure on slide 52 of lecture 4. Note, your code won't look the same as the code on that slide but the order of the sections should be the same. Remember, consistency is key!

2.8 Documentation and Report (30%)

You need to submit User Documentation as well as a Report, both of which should be in a doc/docx or pdf format.

2.8.1 Documentation

Your User Documentation will be a minimum of 2 pages long and should include the following:

- An overview of each of your program's features.
- A guide on how to use your program.
- A discussion of your code, explaining the features you implemented, how you implemented them and why you implemented them the way you did.

The user documentation will be used and referred to by the programmers in the data analysis team at your 'workplace' to get an understanding of your code.

2.8.2 Report

Your report will be a mini-paper that is 2-3 pages long and should follow the structure of a standard academic report. This is what will be used by the data analysis team at your work to make decisions about the future of the company. As a result, this report should discuss the insights you found in the data using your program. Such as, what games have the most entries? What is the distribution of games (by rating) on sales data? Which game has the most highest rating values and/or any other interesting insights you discovered? The required sections are:

- Abstract: Summarise your report's findings. Explain the purpose of the program, the
- questions you wanted it to answer and your outcomes/recommendations.
- Background: Discuss the purpose of the program and your choice of questions (see
- examples in the paragraph above).
- Methodology: Discuss how you went about gaining these insights from the program to answer your questions and why you chose to do it that way. Include commands, input files, and outputs – anything needed to reproduce your results.
- Results: Present the results of your analysis – include tables, plots and discussions.
- Conclusion and Future Work: Give conclusions and what further investigations could be done.
- References: see unit outlines for styling guide.

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

3 Submission Details

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.

3.1 Submission Instructions

You should submit a single file, which should be zipped (.zip). The file must be named FOP_Assignment1_<student id> where the <student id> is replaced by your student id ignoring the angle brackets. There should be no spaces in the file name; use underscores as shown. After creating the zip file, open it and make sure everything zipped correctly! The file must contain the following:

- Your code. This means all the 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 and Report, as described above.

A signed and dated Declaration of Originality. This is available on Moodle and asks you to confirm that your work is your own. 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.

3.2 Unit Passing Requirements

Please note, as specified in the unit outline, it is necessary to have reasonably attempted the assignment in order to pass the unit. **A reasonable attempt is around 30%. Something to note is that this assignment is worth 30% of the overall unit and you need to achieve at least 50%** overall to pass this unit. This means doing poorly in this assignment may make passing difficult. 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.

3.3 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.

3.4 Clarification 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 Moodle 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 section of the unit's Moodle page.

3.5 What Should be Avoided.

- Any unacceptable online source for coding (e.g., ChatGPT, GitHub code, etc).