**Wayne State University**

**Upload Date: Feb 8, 2024**

**CSC 4110 - Software Engineering**

**Weekly Homework**

**Video Link:** [**https://www.youtube.com/watch?v=-t9MotIkS-k**](https://www.youtube.com/watch?v=-t9MotIkS-k)

**Directions:**

**Do all problems by the due date. Follow instructions explicitly. See general requirements for each problem.**

**There are TWO parts to this assignment:**

**CODING and QUESTIONS/ DOCUMENTATION**

**Part One: CODING**

See **General Requirements (pg 5)** - some **Problems have ‘Specific’ requirements**

**Directions:**

* Customer requests to be fulfilled **ON-TIME**
* .py as .txt file to be uploaded to GitHub repository (see below links)
* GitHub link placed in COMMENT section of Upload folder
* Code images and output images placed in THIS original assignment, uploaded to course shell (with appropriate comments, etc…)
* Comments appropriate and explanatory, contain all necessary information

**Problem One**

**\*\*\*\* Adhere to ‘General Requirements’**

**Assignment: Non-disparate DATA WAREHOUSING**

Simulate non-disparate data warehousing with the following **sequence**:

**Step One: ‘data collector’**

Create a **data collector** method, simulating user records with the following attributes: username, password, birthdate, address, social security number,productPurchased,salesperson. This program **procedurally generates ‘sample’ data and stores that data.**

A screenshot of a computer code

Description automatically generated

Figure 1: P1 Code Lines 23,51. This section of the code represents the fake data generator. Lists of probable data are predefined for each category, then random items from each list are selected.

**Step Two: ‘key/value’ pairs**

Feed data collector values into ***key-value pair***. For example, the user data may be an entire list sequence, which is then considered the ‘value’ portion of a unique user ID key.

A screenshot of a computer code

Description automatically generated

Figure 2 – P1 Code lines 52-63. This section of code contains the key system, each key corresponds to the location in the masterlist sublist where the particular data set is stored. This is used for the search engine so that the user can easily type in the category they want to search, which is then converted into the location where the search is to occur.

**Step Three: search engine**

This key-value structure must be searchable. For example, a user may be able to search the entire data store for users in a certain state, or see which users were handled by a certain salesperson (or sales ID).

A computer screen shot of a program

Description automatically generated

Figure 3 – P1 Code Lines 64-86. This section of code contains the search engine, the user simply inputs the category they want to search for, then what in that particular category they want to search through. Pressing enter for either search option simply prints the entire category. The use of key value pairs greatly simplifies the way this search engine is written.

**Problem One Requirements**

“ProductPurchased” consists of order/vendor information such as usernames, web orders, product IDs, quantities, date of order, region, etc.. Any item referring to products, such as Product ID should have the prefix “ID,” such as “ID-trxdfn.”

*The specific design, method and procedure details are up to the student; the column/ category names are up to the student.*

***The student MUST create a DATA COLLECTOR method that PROCEDURALLY generated USER DATA; then the DATA must be placed into key/value pairs and be searchable.***

**ProblemTwo**

**\*\*\*\* Adhere to ‘General Requirements - LAST PAGE’**

**Assignment: Create a ‘Game of Chance’**

Create a ‘game of chance’ simulation to do the following:

(a) build and populate treasure chest with as many items customer requires

(b) create a bank / loot stash

(c) wagers to be placed per “spin” or treasure chest “grab”

(d) customer “plays” until bank account reaches 0 or below.

**Problem Two Requirements:**

Note: the name of the simulation shall be “pirate” related; copy/ paste code and output, showing different outcomes; “**random**” module is to be imported.

A screenshot of a computer program

Description automatically generated

Figure 4 – P2 Code lines 1-23. This section of code demonstrates the creation of the loot chest which is populated using items randomly selected from a predefined master list of items. The player bank account and item winnings are also created.

A screenshot of a computer program

Description automatically generated

Figure 5 – P2 Code lines 25-53. This section of code contains the actual gameplay. The player is asked if they want to play, if they have no money the game automatically ends. The system works by generating a random number in the range to the length of the treasure chest’s inventory. This item is then added to the player’s victory master list and removed from the treasure chest’s inventory list. The bank balance is then reduced, and the loop repeats until the player runs out of money or presses anything other than enter.

A screen shot of a computer

Description automatically generated

Figure 5B – A cool skull is printed out to add visualization. Wow so cool and pirate themed.

**ProblemThree**

**\*\*\*\* Adhere to ‘General Requirements - LAST PAGE’**

**Assignment: Create password simulator**

Customer needs a password simulator to do the following:

(a) create random passwords in perpetuity

(b) if the password is “acceptable,” it gets archived

(c) “unaccepted” passwords get deleted

(d) no less than 40 iterations

**Problem Three Requirements:**

Customer rules of ‘accepted passwords’ include: “special symbols,” and password cannot be a word in a dictionary list; “**random**” module to be imported.

A screenshot of a computer program

Description automatically generated

Figure 6 – Problem 3 code. This code is pretty simple, it generates passwords using randomly selected characters from an acceptable list of characters. These passwords are then checked to see if they fail the password rules. If they fail the rules by not including a special character, then the password is deleted. NOTE: Any password with a special character is inherently not in the dictionary. Note that problem 3 includes user ended sonification where they are encouraged to shout ‘Yay!’.

**General Requirements:**

1. **Add labeling/ comments (name, date, revision #); add in-line requirements where appropriate (such as syntax usage).**

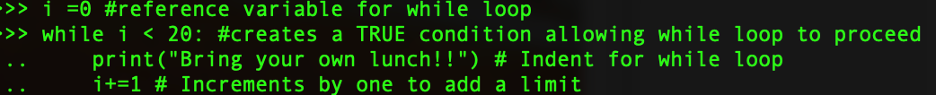
1. **AT LEAST ONE PROBLEM MUST USE SONIFICATION AND VISUALIZATION.**

#Indicate coding begin and end

Example acceptable code comment:

**# Revision number BEGIN/ START DATE**

**## Begin John D. Student here (date)**



**# Revision number FINAL DATE**

**## End John D. Student here**

**# Group / manager/ lead tech/ project # ←-Where appropriate**

1. **Adhere to the following coding style (from PEP8):**

1. Wrap lines so that they don’t exceed 79 characters.

2. Use blank lines to separate functions and classes, and larger blocks of code inside functions

3. When possible, put comments on a line of their own.

4. Where appropriate, name your classes and functions consistently; the convention is to use UpperCamelCase for classes and lowercase\_with\_underscores for functions and methods.

1. **GitHub:**

GitHub Video 1: <https://www.youtube.com/watch?v=fJtyf62yAb8>

GitHub Video 2:<https://www.youtube.com/watch?v=GqNAD4XoZ6k>

**Reference following article to create repository so you can load this assignment output:**

**https://docs.github.com/en/desktop/installing-and-configuring-github-desktop/overview/getting-started-with-github-desktop**

1. **Fill out below CRD**

**Change Request Document**

**Name:**

**Student access ID:**

**Project:**

**Date:**

**Group Number:**

***Everything in italic should be changed as appropriate by you and should not be italic when submitted****. Also remember code is not changed until the Refactoring stage, so don’t put “I changed” or similar until section 4 of the report.*

*(Title of the change request*)

**1.** **Change Request and concepts*:***

*In this section, describe the change that you were requested to do. Give any relevant background information or any essential details.*

*Extract significant concepts and list them here.*

***2.*** ***Sources:*** *Include any sources that you cited or used information from*

***3.*** ***Highlighted Source Code:***

*Attach or cut and paste the code of the classes that you changed. Highlight the code that was changed or added. Use YELLOW for modified code RED for deleted code, and GREEN for added code.*

*If you only changed one method in a large file, only include that method and the file name it’s from. Likewise, if you only changed a line or two in an event map or resource file, only include a few of the surrounding lines and the file name. Do not include thousands of lines of code that you did not change!*

**Part Two: QUESTIONS**

**Answer End of Chapter Questions with Real-Life Examples, documented by APA references, at least ONE reference per Question.**

I’m not creating references. That is a massive waste of time.

See following link to automatically CREATE your references:

<https://owl.purdue.edu/owl/research_and_citation/apa_style/apa_formatting_and_style_guide/general_format.html>

**Question One**

What are the main differences between a ‘data warehouse’ and a typical SQL database?  
  
A data warehouse is for storing data for the sake of storage. A typical SQL database by comparison is meant for transactions. It’s pretty much the difference between an unorganized desk drawer, and a cash register. One is meant for bulk storage, the other is meant for frequent transactions.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Two**

What are differences between someone tracking expenditures via an Excel spreadsheet versus an SQL database? (eg. scale)

A major difference is accessibility, more people are trained on excel vs SQL. This presents a major problem because SQL offers better scaling, but your average user is unlikely to be able to achieve benefits from this because they cannot use it. However, even through Excel can handle hundreds of thousands of rows of data, it cannot handle millions or billions – a database like SQL is needed instead.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Three**

In compiled languages, what steps do programmers do to produce an executable file?

1. Write the code.
2. Compile the code using a compiler (if you installed an IDE, you probably already have this).
3. Link the multiple files together using a linking tool to generate a .exe file.
4. Be frustrated that somehow it didn’t work as intended
5. Repeat steps 1 through 4 until you get it working.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Four**

What is the role of version control systems in software projects?

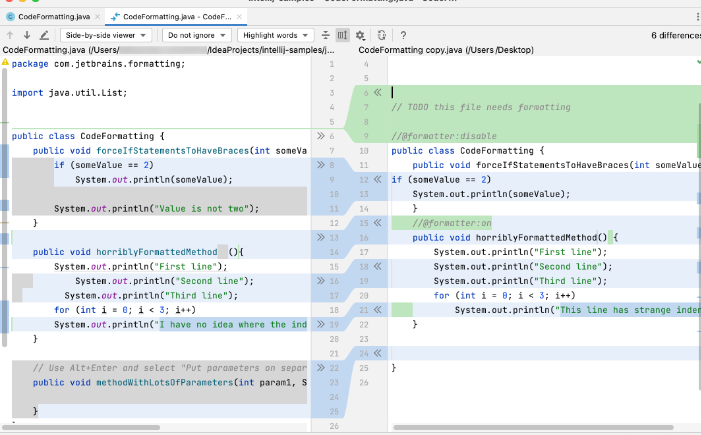
Version control is used so that multiple contributors can make changes to one project at the same time. Version control saves changes to the original version and allows you to merge only the changes – and select which changes you want to merge. This is much more convenient than passing one file around. A good example of version control that is not Github is the version control program for the Fallout 4 and Skyrim creation kits. This version control file system allows changes to .esm (elder scrolls master) files to me add through individual files called .esp (elder scrolls plugin) files. Changes to the .esp files are merged into the master file via version control. Version control however can be extremely finnicky – the inevitable issue is that some user touches something they shouldn’t have, and as a result the person in charge of merging changes has to notice this change or they will otherwise merge it causing things to get messed up. XEDIT is another program for Fallout 4 that achieves many of the same features as the built in Version Control manager – but has a much worse UI so you probably shouldn’t bother.

Give two examples of Github-like applications/ programs and cite their differences.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Five**

What is a ‘**diff**’ file?

Paste a screenshot of a **diff** file here (showing side by side)  
A diff file is a file that shows the different between two files. It is useful for stuff like version control if you hate yourself and don’t want to use the built in GUI that most version control systems tend to come with.   
  


Look at this horrible version change system found online. It’s annoying, I’m glad I’m not a system administrator having to deal with merges.

**Question Six**

Explain what a baseline is.

A baseline refers to an approved software configuration that has already been approved. It is the ‘baseline’ for which future changes are made in version control systems.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Seven**

How does the program version in the private workspace differ from the baseline version?  
  
The program version usually represents the current build whereas the baseline version was a previous stable build.

What is a conflict in terms of two different updates to a file? How does it get resolved?  
  
In my experience conflicts usually get resolved in favor of the more recent change. This isn’t necessarily good and is a pain to parse through to find out where the changes were actually supposed to be made.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Eight**

What is the build and what is the result of the build?  
  
The ‘build’ is the end result of the compilation linking and packaging of the source code into useable software. The end result is usually a .exe file or some library or package.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Nine**

What is the three-tier architecture?  
  
Three tier architecture is the concept of dividing a single application into multiple tiers to be used by different users. The top layer is the User Interface, which is what is used by most users – it’s stuff like a website or application that the user actually interacts with. The next layer down is the Application Tier, where the actual ‘doing’ of the program occurs – the application tier communicates with the user interface to produce results to then display in the user interface. The Data Tier is the last tier and deals with storing data – it basically just communicates with the application tier to deal with data storage.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

**Question Ten**

What is polymorphism in technology? Give an example.  
  
Polymorphisms are ‘many forms’, they’re usually boring stuff like something having the same name but behaving differently based on the context used. Method overriding is fairly common to achieve polymorphisms.

Astley, R. (2009, October 25). *Rick Astley - Never Gonna Give you up (official music video)*. YouTube. https://www.youtube.com/watch?v=dQw4w9WgXcQ

NOTE:

