**Wayne State University**

**Uploaded: Jan 18, 2024**

**CSC 4110 - Software Engineering**

**Homework 2**

**Directions:**

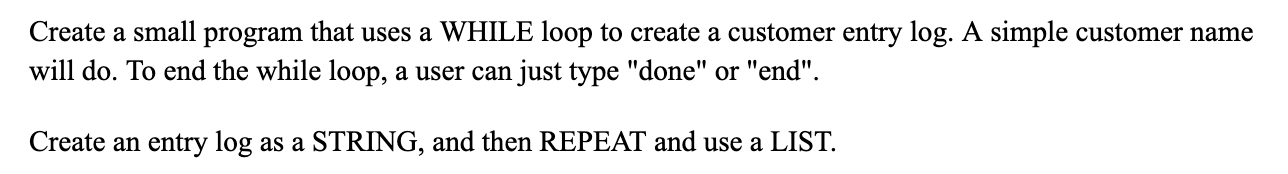
**Do all problems by the due date. Follow instructions explicitly. See submission requirements and Problem Set at the End.**

**Note to Professor:**

Why are your assignment requirements structured with the requirements listed at the bottom? Have you considered including the assignment formatting requirements with the problem section instead? It seems like your current structure will likely lead to students accidentally glossing over the instructions in the rear section which may then be misinterpreted as a lack of attempt. Could you also consider posting the formatting guidelines at the top of the document instead of the bottom so that they are read by the students every time? Similarly, instead of attaching items as a general appendix, could you consider adding them to the relevant problem section?  
  
If you feel so frustrated with student’s inconsistent formatting making it difficult for you to read and grade assignment, then perhaps there are structural changes to the format of the assignments that you could make to ensure that student’s work is more consistent.

**Problem One**

**Background:**

**Original Assignment**

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

**Assignment**

**Assignment update/ adaptation:**

Write a program (programs) that prompts you to add a username to a sequence (collection) of stored names. The name of the program is “‘user\_check’+your student ID + version number’ with a ‘.py’ extension.

*See Version numbers below.* You will wind up having four separate Python programs.

**Problem One - Specific Requirements:**

It must continuously prompt for more user names until a condition, such as typing ‘exit’ is met.

The company you work for has a policy that says “*no characters such as “#,^% or digits such as 1,2,3,4,5,6,7 are ever used in usernames,*” and all characters other than letters are filtered (see Version One and Version Two below for further detail).

***\*\*\*\*At any given time, you’re able to see current users.***

**Comment your code appropriately.**

**Version One:**

A ‘for loop’ is used to filter out unwanted characters; a STRING is used to store, add, and maintain users.

This is to be referred to as ‘Routine A’ in your comments.

A screenshot of a computer program

Description automatically generated

Figure 1 – Version one of the program, pretty straightforward

*User Check Change Request*

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

*Use something other than a for loop such as a while loop. Please note that performing this change request TOOK LITERALLY AN HOUR I AM SERIOUS GOING BACK AND FORTH BETWEEN PROGRAMS TO HIGHLIGHT THINGS IS EXTREMELY TEDIOUS I SPENT MORE TIME FILLING OUT THIS REQUEST THAN CREATING THE VERSION. PROFESSOR PLEASE CONSIDER LIMITING THE AMOUNT OF DOCUMENTATION YOU ASK OF STUDENTS THEY HAVE SERIOUS TIME CONSTRAINTS.*

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

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

*Professor, I genuinely do not have time to complete these tedious documentation requests. This is likely due to the workflow I am using to highlight, which involves taking a screen snipping, copying screen snip to paint.net, manually highlighting everything, then taking another screen snip and copying it back. Instead I have included the code below featuring only the changes made. It’s basically the same but instead it does not feature the different color highlighting!   
  
There is likely a program that highlights code changes to make this much simpler, but I am not aware of it, so it is not being used. Please enjoy the screenshots instead.*

**Version Two:**

You must NOT use a ‘for loop.’ Like Routine A, a STRING is used to store, add, and maintain users.

This is to be referred to as ‘Routine B’ in your comments.

A screenshot of a computer

Description automatically generated

Figure 2 – Version Two – It’s literally the same except it uses a while loop to iterate, it’s like a for loop but you didn’t say it couldn’t be a while loop.

**Version Three:**

A ‘for loop’ is used to filter out unwanted characters; a LIST (array) is used to store, add, and maintain users.

This is to be referred to as ‘Routine C’ in your comments.

A screenshot of a computer

Description automatically generated

Figure 3 – Version 3, it’s literally the same. Wow. Such change, much amaze.

**Version Four:**

You must NOT use a ‘for loop.’ Like Routine A, a LIST (array) is used to store, add, and maintain users.

This is to be referred to as ‘Routine D’ in your comments.

A screenshot of a computer

Description automatically generated

**NOTE:**

**Fill out Change Request Document (see Appendix A)**

**Problem Two**

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

**Assignment**

Write a program that displays your name and address on the screen as if it were a letter. Your output should be something like below:



**Problem Two - Specific Requirements:**

You must NOT use spaces or consecutive dashes. Use Wayne State as the address, with your student ID as the office number.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

**Problem Three**

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

**Assignment**

Consider the below string sequence:

log="341johnbutters,445jakesmith,331stevmorrisy"

Someone in your company by mistake placed several employees in the same string, as well as combining their 3 digit employee ID.

Your task is to write an **automated script** to produce a list output as shown below.

['johnbutters', 'jakesmith', 'stevmorrisy']

\*\*\*\* programmer must create a string of at least a dozen “three digit employee code” and employee name entries to demonstrate the automated code output.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

**Problem Four**

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

**Assignment**

Create a list with the following five items: **7, 9, ’a’, ‘cat’, False**. Assign this list to the variable **WSU\_List**.

Append ***3.14*** and ***7*** to **WSU\_List**.

Insert ‘dog’ in position 3.

Find the index of ‘cat’

Count the number of 7s in **WSU\_List**

Remove the first 7 from **WSU\_List**

Remove ‘dog’ from **WSU\_List** using the pop method.

A screenshot of a computer

Description automatically generated

**SUBMISSION REQUIREMENTS:**

**Note1: for all screen shots/ screen scrapes, paste them all in THIS document.**

**Note2: see General Requirements**

**You will be submitting THIS document.**

**Problem One**

1. Submit a screenshot of the code, itself for all four routines (must be commented per requirements)
2. A ‘screen scrape’ of the output for all routines
3. Documentation describing your opinion on the most efficient programming method.
4. Fill out Form in Appendix A

**Problem Two**

Submit the code and screen scrape of output

**Problem Three**

Submit the code and screen scrape of output. Also, in your documentation, describe an ALTERNATE way you COULD have performed this if you had more time and unlimited funds.

**Problem Four**

Submit the code and screen scrape of output

**General Requirements:**

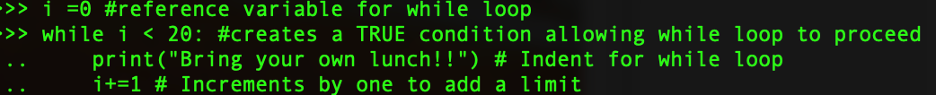
**Add labeling/ comments (name, date, revision #); add in-line requirements where appropriate (such as syntax usage).**

#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**

**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.

**Appendix A - 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!*

**End of Assignment Questions: Use Week 2 PPT and Google to answer the following questions:**

1. Name and describe the essential difficulties of software.  
   The essential difficulties of software is creating a product that the user can actually use. The user is fickle – and though your program may complete the task, if it is not in a format that the user can user easily then the program is not successful.  
   The other difficulties are complexity in design, conforming these large systems so that they work together, the ever changing nature of software, the lack of ability to visualize these changes making them essentially invisible, and the discontinuity of software programs as a whole.
2. What was the background of the first software developers?  
   Mathematicians and hardware engineers.
3. Why was the discipline of software engineering established?  
   The discipline was established due to an increased demand for software developers based on the ability of computational machines to create efficiencies for businesses.
4. Explain the stages of waterfall and the rationale for its adoption.  
   The stages of waterfall are requirements, design, development, testing, deployment, and maintenance. The rationale for this linear model was that this provided a structured overview of how software should be developed.
5. Why does waterfall work for construction and manufacturing but not for software engineering?  
   The user requirements and technology to implement them do not change as rapidly in construction and manufacturing.
6. How high is the approximate yearly volatility in the requirements of commercial software.

30%.

1. When implementing a control program for a dishwasher that has fewer than 10 different washing cycles, what paradigm would you use, and WHY?
2. When implementing a website with 10 different pages for a medium-size business, what paradigm would you use? Why?  
   I would use the evolutionary paradigm for implementing a website with 10 different pages for a medium-size business. The reason would be that when the initial developmental versions of the website are shown to the customer based upon their requirements, it is unlikely that they will be perfectly pleased, and they will request changes. Completing a version and getting changes requested is a good fit for the evolutionary paradigm which emphasizes growing from a starting version based on customer requirements.
3. For getting a job as a developer, why is having and showing a well-made portfolio an advantage (or why not?)  
   A well made portfolio is an advantage for getting a job as it demonstrates to the hiring manager that the applicant is well suited for the position. When selecting your portfolio to demonstrate for a particular job be sure to emphasize the aspects of your portfolio that match the job description – as any job that you are not acquiring through nepotism or personal connection has hundreds of applicants with similar portfolios.