Project: Designing a Python Solution

**Purpose**

* Design a computational artifact that solves a meaningful problem.
* Use design thinking and problem-solving skills to plan before you code.
* Apply core programming concepts in your pseudocode to build something engaging, accessible, and functional
* Share your processes, ideas, and results clearly and effectively on the problem and your solution design.

**Points:** 50  
**Effort:** Individual work, collaborative discussion encouraged [CS Academic Integrity and AI Policy - Harding](https://docs.google.com/document/d/1SY4-RMJ2B9GkEbTzmpRVSfKKsgQnYWYF/edit)

You are encouraged to collaborate to discuss concepts but individually write your own code and answer the questions in your own words . When you ask questions and explain to others you get a deeper understanding. You can use AI tools but remember

* + Acceptable: Using ChatGPT to explain something in python
  + Unacceptable: Asking ChatGPT to generate your design or explanations

**Due Date:**  See Canvas.

Work will be accepted up to 24 hours after the due date with a 10% penalty. Meaning if you turn it in at 12:01 am of the next day you will be deducted 10% of the total points from your score. If the assignment is more than 24 hours late, it will be a 0.

**Deliverables:** Upload this document as a pdf or word document with your answers in yellow boxes and make sure your images are cropped and big enough to read.

1. Final Project Requirements

2 Project Overview

3. Design Main program and Functions

[3.1 Data](#_c7roaxcljocb)

[3.2 Break Problem into Smaller Pieces](#_l41popvpiqux)

[3.3 Design Flow of Main](#_8z6cmic6br7h)

[3.4 Functions](#_e73wo1eibf85)

[4 Reflection](#_35aww0ax5trf)

# 1. Final Project Requirements

Your Python program must include:

* A brief summary of what the program does
* Use of the user’s name to personalize interaction
* At least one meaningful calculation
* A closing message
* At least four variables, including both string and a numeric type (int or float)
* At least one list used to store or process data
* At least two different user inputs, including one string and one number
* Decision-making using if statements
* Both for and while loops
* Two or more functions, including at least:
  + One function that takes parameters
  + One function that returns a value

# 2 Project Overview

**2.1 Problem and Solution**

|  |
| --- |
| 2.1 Write Problem Statement from last assignment   * **All users** who have online accounts need at least a basic, free, and easy-to-use application for managing account information. * A User can use a password manager whenever they get on their computer and still keep their information secure * CISA standards are unenforceable, and 3 out of 4 websites may not already enforce password creation standards. With this, most users who do not know these standards don’t use them. Users are more than likely not forced to make a good password and therefore don’t know what a good standard looks like.   Source: https://research.gatech.edu/largest-study-its-kind-shows-outdated-password-practices-are-widespread |
| 2.2 How will your program solve the problem? What will it do?  My program will generate passwords to fulfill the requirement that they be complex. My program will store passwords and encrypt saved data. The user will need to create and save a key to access these files. The goal is that the password manager is simple to use and easy to access. |
| 2.3 List 3–5 features of your program.   * Use a GUI (eliminate user requirement to use a python terminal) * Create and store passwords * Utilize encryption to protect plaintext data from being read easily. |

# 3. Design Main program and Functions

Review the project requirements to make sure you include them in your design

## 3.1 Data

Describe the data that will be stored in your program

|  |
| --- |
| 3.1.1 You must use at least one list.  What will your list store?   * Randomly created passwords/characters (for generating). The list for passwords will use **.join** to make whole words (new password). |
| 3.1.2 Describe variables you will need and their data type.   * I will need variables that store string items(characters) in their own lists. These items will be in lists containing specific characters and will be pulled by a function and added to an empty **list** (new list of random characters). They will then be joined into a password. Every password will be added to a seperate list of saved passwords. This function will loop to create several passwords. |

## 3.2 Break Problem into Smaller Pieces

Break the problem into smaller tasks. Describe how the user will interact with your program and what the program will do.

* What user does
* What program does

|  |
| --- |
| List the tasks your program will do.  A user will see the GUI buttons and understand that the program can create passwords, store passwords, and provide information about passwords.   * Option 1 will generate passwords. This can create one or several (for several accounts). Option for numbers, letters, and special characters or numbers and letter only, will be provided. * Option 2 will have a function to store and list saved passwords. This option will use read/write methods to make .txt files. * Option 3 will list references and resources to give the user an understanding about password safety. This is only an output. |

## 

## 3.3 Design Flow of Main

Create pseudocode to show how your main program will flow.

Pseudocode expectations: Write your pseudocode on paper, whiteboard or tablet and upload a photo. Do not type and do not use AI to produce your pseudocode. Pseudocode code more like English and include the control structures (if/else, for, while)

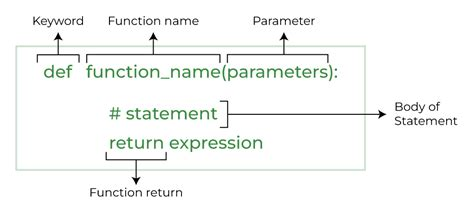
|  |
| --- |
| 3.3.1 Create your pseudocode for the main program. Upload a photo or scanned copy. (Do not type)  Main Menu Pseudo Code:  Flow Chart: |

## 3.4 Functions

Think about the tasks above. Remember the requirement to include two or more functions that

* At least One function that takes parameters
* At least one function that returns a value

Describe at least 2 functions you will write to do tasks in your program. What will each function do? What will the function parameters be? What will the return values be? For each function write the method name, parameters, return value and pseudocode for the function.



|  |
| --- |
| 3.4.1 What is the purpose of the function?  This function is the logic of how random passwords are created. This function will run based off of user inputs and create random strings. The strings are joined and create the password.  Function name: **password\_gen1()**  Parameters: none  Return Value: new password  Put image of your algorithm as pseudocode below |

|  |
| --- |
| 3.4.2 What is the purpose of the function?  The previous function is the logic of the password generation and it was written for a python terminal. This function is the GUI aspect of getting inputs from the user. This will ensure only usable information is retrieved from the user. The input information is passed off the to the password generator.  Function name: **gen\_button(user\_input1, use\_input2)**  Parameters: **(user\_input1, use\_input2)**  Return Value: **gen\_function(char\_count, password\_count)**  Put image of your algorithm as pseudocode below |
| 3.4.3 What is the purpose of the function?  Function name:  Parameters:  Return Value  Put image of your algorithm as pseudocode below |

# 4 Reflection

|  |
| --- |
| 4.1 What challenges do you anticipate during implementation?  The encryption/decryption features and Tkinter utilization is new for me.  The tkinter is slightly convoluted as a topic and it feels as if I am learning a new language. This is comparable to learning how to use bootstrap in html. The aspect I am trying to get used to is making labels and understanding how to manipulate the grid system.  The encryption/decryption implementation is new to me and has already required troubleshooting just to install the cryptography library. I had to reinstall python to circumvent my problem. The methods included in this library are new topics but they are not too complicated to implement. |
| 4.2 What would be an additional goal or advanced feature if you had more time?  I would want to understand the different types of file encryption that exist. I was unaware of the specific standards of encryption that exist with more popular password managers. I do not know if newer methods are harder to implement but I will start simple and branch off what I understand. |
| 4.2 How could your program affect users or communities in a positive way?  Users can have a free to use password manager that is easily accessible without internet connection. The goal is to create a simple but reliable program with minimal barriers. If I can get one person to move away from writing accounts and passwords in a notebook (like some family members) than I would feel great about that. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Exemplary (A)** | **Proficient (B)** | **Developing (C–D)** | **Incomplete (F)** |
| 1. Design Requirements Met  (10 pts) | All required elements are included: use of name, list, 4 variables, loops, decision-making, 2+ functions (with parameter/return), and a calculation. Design clearly supports problem solution. | One or two required elements are weak, unclear, or missing. Design mostly aligns with problem. | Several required elements are vague or not present. Design is not well-aligned to the stated problem. | Major requirements are missing or not addressed. Program idea is unclear. |
| 2. Program Design: Data, Main Program, and Function Design  (20 pts) | Data (list, variables) is thoughtfully chosen and well-matched to program. At least 2 functions are clearly explained with purpose, parameters, return values, and pseudocode. Pseudocode is uploaded and matches goals. | Data and function plans are mostly clear, but may have small gaps. At least one function may lack parameter/return info or the pseudocode may be incomplete. | Design is unclear or missing key elements. Pseudocode is missing or not original. Function purpose or structure may be incorrect. | Little to no function planning, pseudocode, or data description provided. |
| 3. Project Overview and Reflection  (10 pts) | Problem and solution are clearly explained. Features are well thought-out. Reflection includes thoughtful discussion of challenges, readability, stretch goals, and potential impact. | Overview and reflection mostly complete. Impact may be superficial or stretch goal not well described. | Reflection and overview are vague or incomplete. Problem or solution may be unclear. | Overview or reflection missing. No effort to connect to impact or deeper learning. |