Wayne State University
Upload Date: Feb 8, 2024
CSC 4110 - Software Engineering
Weekly Homework

Video Link: 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.**

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.

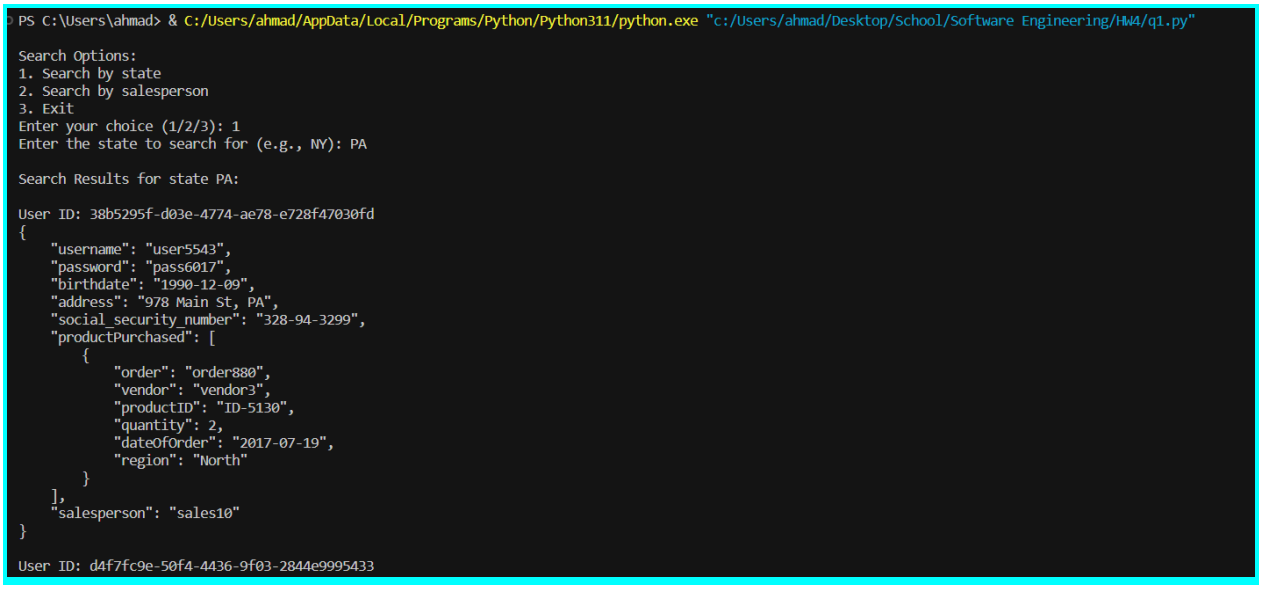
Step Three: search engine

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).

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.



User ID: d4f7fc9e-50f4-4436-9f03-2844e9995433


```
{
    "username": "user6600",
    "password": "pass5727",
    "birthdate": "1978-12-13",
    "address": "514 Main St, PA",
    "social_security_number": "302-69-1170",
    "productPurchased": [
        {
            "order": "order328",
            "vendor": "vendor3",
            "productID": "ID-8265",
            "quantity": 4,
            "dateOfOrder": "2016-09-16",
            "region": "North"
        },
        {
            "order": "order114",
            "vendor": "vendor3",
            "productID": "ID-2976",
            "quantity": 5,
            "dateOfOrder": "2017-01-31",
            "region": "South"
        },
        {
            "order": "order647",
            "vendor": "vendor8",
            "productID": "ID-8131",
            "quantity": 3,
            "dateOfOrder": "2018-08-27",
            "region": "West"
        }
    ],
    "salesperson": "sales8"
}
```


User ID: d959e4d5-1033-400c-b18b-68f632e076f2


```
{
    "username": "user3001",
    "password": "pass1496",
    "birthdate": "1955-09-10",
    "address": "737 Main St, PA",
    "social_security_number": "721-91-2937",
    "productPurchased": [
        {
            "order": "order463",
            "vendor": "vendor7",
            "productID": "ID-2816"
```

```
            "region": "North"
        }
    ],
    "salesperson": "sales5"
}

User ID: 43099cad-8244-4cc1-8873-8a044f27c087
{
    "username": "user6081",
    "password": "pass5984",
    "birthdate": "1958-05-23",
    "address": "707 Main St, PA",
    "social_security_number": "309-20-2251",
    "productPurchased": [
        {
            "order": "order985",
            "vendor": "vendor4",
            "productID": "ID-8509",
            "quantity": 2,
            "dateOfOrder": "2020-05-25",
            "region": "East"
        },
        {
            "order": "order145",
            "vendor": "vendor9",
            "productID": "ID-7153",
            "quantity": 2,
            "dateOfOrder": "2018-03-27",
            "region": "South"
        },
        {
            "order": "order943",
            "vendor": "vendor4",
            "productID": "ID-2797",
            "quantity": 5,
            "dateOfOrder": "2019-07-03",
            "region": "North"
        }
    ],
    "salesperson": "sales10"
}

Search Options:
1. Search by state
2. Search by salesperson
3. Exit
Enter your choice (1/2/3):
```

```
import random
from datetime import datetime, timedelta
import uuid
import json

# Enhanced Data Collector with detailed product purchase information
def generate_random_date(start_year=1990, end_year=2020):
    start_date = datetime(start_year, 1, 1)
    end_date = datetime(end_year, 12, 31)
    time_between_dates = end_date - start_date
    random_number_of_days = random.randrange(time_between_dates.days)
    return (start_date +
timedelta(days=random_number_of_days)).strftime('%Y-%m-%d')

def generate_random_address():
    states = ['NY', 'CA', 'FL', 'TX', 'PA']
    return f"{random.randint(100, 999)} Main St, {random.choice(states)}"

def generate_product_purchased():
    products = [
        {"order": f"order{random.randint(100,999)}", "vendor":
f"vendor{random.randint(1,10)}",
         "productID": f"ID-{random.randint(1000,9999)}", "quantity":
random.randint(1,5),
         "dateOfOrder": generate_random_date(2015, 2020), "region":
random.choice(["North", "South", "East", "West"])}
        for _ in range(random.randint(1,3))  # Each user can have multiple product
purchases
    ]
    return products

def data_collector(num_records):
    users = []
    for _ in range(num_records):
        user = {
```

```
            "username": f"user{random.randint(1, 10000)}",
            "password": f"pass{random.randint(1000, 9999)}",
            "birthdate": generate_random_date(1950, 2000),
            "address": generate_random_address(),
            "social_security_number": f"{random.randint(100,
999)}-{random.randint(10, 99)}-{random.randint(1000, 9999)}",
            "productPurchased": generate_product_purchased(),
            "salesperson": f"sales{random.randint(1, 10)}"
        }
        users.append(user)
    return users

# Key/Value Pair Storage
def store_data(users):
    user_store = {}
    for user in users:
        user_id = str(uuid.uuid4())  # Generating a unique ID for each user
        user_store[user_id] = user
    return user_store

# Advanced Search Engine
def search_users(data_store, search_type, search_value):
    results = []
    for user_id, user_data in data_store.items():
        if search_type == 'state':
            address_state = user_data['address'].split(", ")[1]
            if search_value == address_state:
                results.append((user_id, user_data))
        elif search_type == 'salesperson':
            if user_data['salesperson'] == search_value:
                results.append((user_id, user_data))
    return results

# Neatly print search results
def print_search_results(results):
```

```
    for user_id, user_data in results:
        print(f"\nUser ID: {user_id}")
        print(json.dumps(user_data, indent=4))

# Main execution with interactive search
if __name__ == "__main__":
    # Generate and store user data
    num_records = 10
    users = data_collector(num_records)
    user_store = store_data(users)

    while True:
        # Ask the user what type of search they want to perform
        print("\nSearch Options:")
        print("1. Search by state")
        print("2. Search by salesperson")
        print("3. Exit")
        choice = input("Enter your choice (1/2/3): ")

        if choice == '1':
            search_state = input("Enter the state to search for (e.g., NY): ")
            search_results = search_users(user_store, 'state', search_state)
            print(f"\nSearch Results for state {search_state}:")
            print_search_results(search_results)

        elif choice == '2':
            search_salesperson = input("Enter the salesperson to search for (e.g.,
sales1): ")
            search_results = search_users(user_store, 'salesperson', search_salesperson)
            print(f"\nSearch Results for salesperson {search_salesperson}:")
            print_search_results(search_results)

        elif choice == '3':
            print("Exiting search...")
            break
```

```
    else:
        print("Invalid choice. Please enter 1, 2, or 3.")
```

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.

```
import random

def interactive_pirate_loot_game(initial_bank):
    # Items in the treasure chest with their corresponding values
    treasure_chest = {
        'Gems': 10,      # Value of +10
        'Debloons': 50,  # Value of +50
        'Coal': -10,     # Value of -10
        'Dirt': -50      # Value of -50
    }

    bank_account = initial_bank  # Initial bank / loot stash
```

```
    print("Welcome to the Pirate's Game of Chance")
    print(f"Ye start with a bank account of {bank_account} coins.")
    print("In ye chest, there be Gems, Debloons, Coal, and Dirt.")

    while bank_account > 0:
        # Prompt the user to grab from the chest
        user_input = input("would ye like to grab something from the chest? (yes/no):
").lower()
        if user_input == "yes":
            item_chosen = random.choice(list(treasure_chest.keys()))
            value = treasure_chest[item_chosen]
            bank_account += value
            if value > 0:
                print(f"Ahoy me matey! Ye found {item_chosen} worth {value} coins.")
            else:
                print(f"Barnacles! Ye found {item_chosen}. That's {value} coins.")
            print(f"Ye now have {bank_account} coins.")
        elif user_input == "no":
            print("Arr! Take yer winnings elsewhere")
            break
        else:
            print("That's not a command ye scallywag!")

        if bank_account <= 0:
            print("Yer out of luck matey!")
            break

# Initial conditions
initial_bank = 100  # Starting amount in the bank account

# Run the game
interactive_pirate_loot_game(initial_bank)
```

```
PS C:\Users\ahmad> & C:/Users/ahmad/AppData/Local/Programs/Python/Pytho
Welcome to the Pirate's Game of Chance
Ye start with a bank account of 100 coins.
In ye chest, there be Gems, Debloons, Coal, and Dirt.
would ye like to grab something from the chest? (yes/no): yes
Ahoy me matey! Ye found Gems worth 10 coins.
Ye now have 110 coins.
would ye like to grab something from the chest? (yes/no): yes
Barnacles! Ye found Coal. That's -10 coins.
Ye now have 100 coins.
would ye like to grab something from the chest? (yes/no): yes
Ahoy me matey! Ye found Gems worth 10 coins.
Ye now have 110 coins.
would ye like to grab something from the chest? (yes/no): yes
Barnacles! Ye found Coal. That's -10 coins.
Ye now have 100 coins.
would ye like to grab something from the chest? (yes/no): no
Arr! Take yer winnings elsewhere
PS C:\Users\ahmad>
```

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

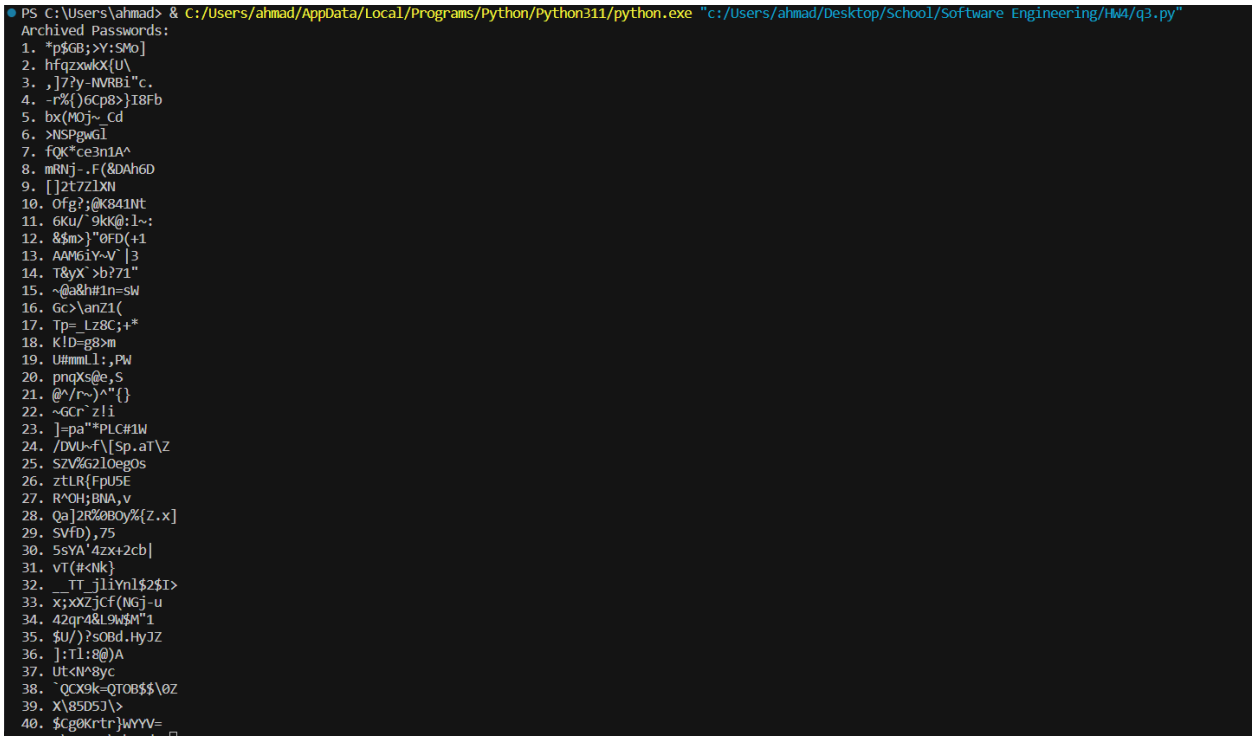
(b) if the password is "acceptable," it gets archived

(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.




```
import random
import string

# Simple dictionary list for demonstration purposes
dictionary_list = ['password', '123456', '123456789', 'qwerty', 'abc123', 'monkey', '1234567', 'letmein']

# Function to generate a random password
def generate_password():
    length = random.randint(8, 16)  # Choose a random length for the password
    characters = string.ascii_letters + string.digits + string.punctuation  # Pool of characters to choose from
```

```
    password = ''.join(random.choice(characters) for i in range(length))  # Generate
a random password
    return password

# Function to check if the password is acceptable
def is_acceptable(password):
    if any(char in string.punctuation for char in password) and password not in
dictionary_list:
        return True
    else:
        return False

# Main loop to generate and archive acceptable passwords
archived_passwords = []
iterations = 40  # Minimum number of iterations

for _ in range(iterations):
    password = generate_password()
    if is_acceptable(password):
        archived_passwords.append(password)  # Archive the password if it's
acceptable

# Display the archived passwords in a neat manner
print("Archived Passwords:")
for i, password in enumerate(archived_passwords, start=1):
    print(f"{i}. {password}")
```

General Requirements:

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

(2) 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)

```
>> i =0 #reference variable for while loop
>> while i < 20: #creates a TRUE condition allowing while loop to proceed
..     print("Bring your own lunch!!") # Indent for while loop
..     i+=1 # Increments by one to add a limit
```

Revision number FINAL DATE
End John D. Student here
Group / manager/ lead tech/ project # **←−Where appropriate**

(3) 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.

(4) 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

(5) Fill out below CRD

Change Request Document

Name: Ahmad Shah
Student access ID: hd6992
Project: Assignment 4
Date: 2/15
Group Number: N/A

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:**

I needed to modify the code so that it can search for the users and display the corresponding information

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!

```
ef search_users(data_store, search_type, search_value):
    results = []
    for user_id, user_data in data_store.items():
        if search_type == 'state':
            address_state = user_data['address'].split(", ")[1]
            if search_value == address_state:
                results.append((user_id, user_data))
        elif search_type == 'salesperson':
            if user_data['salesperson'] == search_value:
                results.append((user_id, user_data))
    return results
```

Part Two: QUESTIONS

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

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?
Give examples and references.
A data warehouse and a typical SQL database serve different purposes and have distinct characteristics. A data warehouse is designed for analytical queries and reporting, optimized for storing and analyzing large volumes of historical data from various sources. It's structured to support complex queries for decision-making and business intelligence. In contrast, a typical SQL database is transactional, optimized for handling transactions efficiently, ensuring data integrity, and supporting real-time operations. While both use SQL for querying, their architectures, optimization strategies, and usage patterns differ significantly to meet their respective needs.

Database vs. Data Warehouse: A comparative review. (n.d.).
https://www.healthcatalyst.com/wp-content/uploads/2014/05/Database-vs-Data-Warehouse-A-Comparative-Review.pdf

Question Two

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

Tracking expenditures via an Excel spreadsheet versus an SQL database differs in several aspects, primarily in scale, functionality, and collaboration capabilities. Excel is suitable for small to medium-scale tracking, offering flexibility and ease of use for individual or small team purposes. However, it becomes cumbersome and prone to errors as data volume increases. On the other hand, an SQL database excels in managing large volumes of data efficiently, providing robust data integrity, security, and scalability. It allows for more sophisticated querying, analysis, and reporting capabilities, making it suitable for complex business environments and facilitating collaboration among multiple users simultaneously.

Google. (n.d.). Data Analysis using SQL and Excel. Google Books. https://books.google.com/books?hl=en&lr=&id=7YWbCgAAQBAJ&oi=fnd&pg=PR33&dq=What%2Bare%2Bdifferences%2Bbetween%2Bsomeone%2Btracking%2Bexpenditures%2Bvia%2Ban%2BExcel%2Bspreadsheet%2Bversus%2Ban%2BSQL%2Bdatabase%3F%2B%28eg.%2Bscale%29&ots=4hpMM8CrOB&sig=gpkrc87Sg8R2-37AIClUm5NU76E#v=onepage&q&f=false

Question Three

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

Give an example.

In compiled languages, programmers typically follow a set of steps to produce an executable file. For instance, let's consider the process for compiling a C program. First, the programmer writes the C code using a text editor. Then, they use a compiler like GCC (GNU Compiler Collection) to translate the C code into machine code. They do this by running a command like "gcc -o program_name source_file.c". This command invokes the compiler, which generates an object file

from the source code. Finally, the linker links the object file with any necessary libraries to create the final executable file, which can be run directly on a computer.

Nanz, S., & Furia, C. A. (2015). A Comparative Study of Programming Languages in Rosetta Code. In 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (pp. 778-788). Florence, Italy. DOI: 10.1109/ICSE.2015.90. Keywords: Java;Programming;Indexes;Statistical analysis;Runtime;Standards.

Question Four

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

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

Version control systems play a crucial role in software projects by managing changes to source code, documents, and other files. They provide a centralized platform for developers to collaborate, track modifications, and maintain a history of revisions. Version control systems enable team members to work concurrently on different aspects of a project, merge their changes seamlessly, and revert to previous versions if necessary. Additionally, they facilitate code reviews, streamline the release process, and help ensure the stability and integrity of the project.

Two examples of GitHub-like applications are GitLab and Bitbucket. GitLab is an open-source platform that offers both self-hosted and cloud-hosted solutions for version control, issue tracking, and continuous integration/continuous deployment (CI/CD). It provides a robust set of features, including code review tools, issue boards, and a built-in Docker registry. Bitbucket, on the other hand, is a product of Atlassian that offers similar functionalities to GitHub and GitLab. It integrates seamlessly with other Atlassian products like Jira and Confluence, making it a preferred choice for teams already using Atlassian's ecosystem. One key difference between GitLab and Bitbucket is their pricing models: GitLab offers a more flexible pricing structure with free, self-hosted, and cloud-hosted options, while

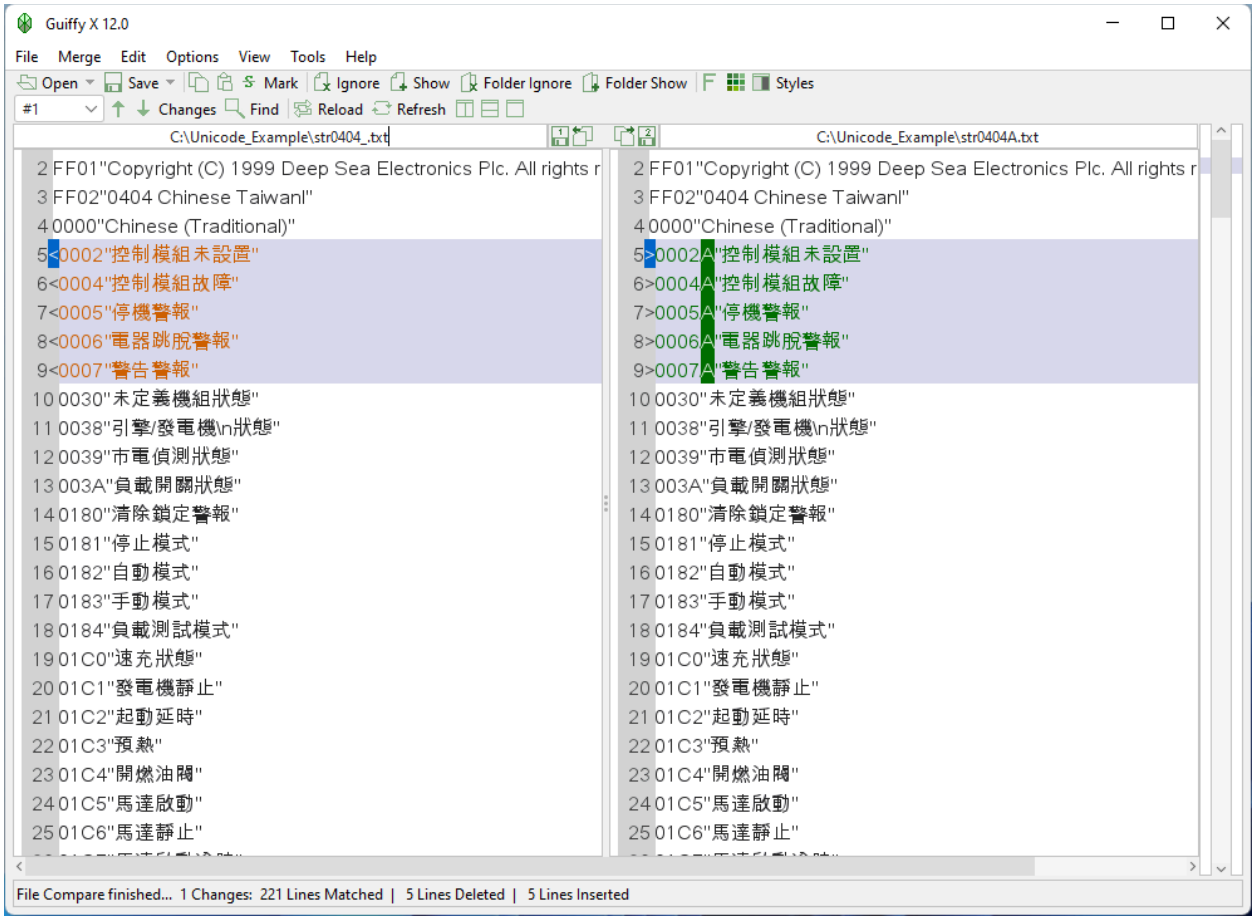
Bitbucket limits its free tier and charges based on the number of users or repositories for its cloud-hosted version.

Google. (n.d.-b). Version control with git. Google Books. https://books.google.com/books?hl=en&lr=&id=aM7-Oxo3qdQC&oi=fnd&pg=PR3&dq=What%2Bis%2Bthe%2Brole%2Bof%2Bversion%2Bcontrol%2Bsystems%2Bin%2Bsoftware%2Bprojects%3F&ots=3btkHDYcsc&sig=16SQNZfLrupjzNyUMH8RGkvDv2Y#v=onepage&q=What%20is%20the%20role%20of%20version%20control%20systems%20in%20software%20projects%3F&f=false

Question Five

What is a '**`diff`**' file?
Paste a screenshot of a **`diff`** file here (showing side by side)

A 'diff' file, short for "difference," is a text file that contains the line-by-line differences between two versions of a file or set of files. It's generated by a diff utility, which compares the contents of two files and identifies the changes made, whether they involve adding, modifying, or deleting lines of text. 'Diff' files typically use a standard format, such as Unified Diff Format (unidiff), which includes metadata such as file names, timestamps, and line numbers to provide context for the changes. These files are commonly used in version control systems to track modifications between different versions of code or documents, facilitating collaboration, code reviews, and merging of changes.



Comparing and merging files with GNU Diff and Patch. (n.d.-a). https://azrael.digipen.edu/~mmead/www/docs/DiffAndPatch.pdf

Question Six

Explain what a baseline is.

A baseline refers to a predefined, stable state or version of a project, typically used as a reference point for comparison or as a foundation for further development. It represents a snapshot of the project at a specific point in time, capturing its key features, functionalities, and configurations. Baselines are commonly established at significant milestones in a project's lifecycle, such as the completion of a major phase or the release of a stable version. They serve as a benchmark for evaluating progress, assessing changes, and managing project scope and quality. Baselines

help ensure consistency, traceability, and reproducibility in software development, allowing teams to effectively track and control variations throughout the project.

Horowitz, M., Joch, A., Kossentini, F., & Hallapuro, A. (2003). H.264/AVC baseline profile decoder complexity analysis. IEEE Transactions on Circuits and Systems for Video Technology, 13(7), 704-716. DOI: 10.1109/TCSVT.2003.814967. Keywords: Automatic voltage control; Decoding; Hardware; Bit rate; Frequency estimation; Computational complexity; Video codecs; Performance analysis; Frequency measurement; Time measurement.

Question Seven

How does the program version in the private workspace differ from the baseline version?
What is a conflict in terms of two different updates to a file? How does it get resolved?

The program version in the private workspace differs from the baseline version by reflecting the changes made by individual developers during their work. These changes may include bug fixes, new features, or optimizations. In contrast, the baseline version represents a stable and agreed-upon state of the project, often corresponding to a milestone or release. While the baseline version serves as a reference point for the project's progress, the private workspace version incorporates ongoing development efforts.

A conflict arises when two different updates are made to the same file, resulting in incompatible changes that cannot be automatically merged. This typically occurs in collaborative environments where multiple developers are working on the same codebase concurrently. Conflicts may arise when developers modify the same lines of code or when their changes affect overlapping sections of the file. To resolve conflicts, developers need to review the conflicting changes manually and decide how to reconcile them. This may involve editing the code to incorporate both sets of changes, discarding one set of changes, or finding a compromise. Version control systems often provide tools to assist in conflict resolution, such as visual diff tools and merge conflict markers, to streamline the process.

Google. (n.d.). Software engineering. Google Books. https://books.google.com/books?hl=en&lr=&id=aK70_4xevYsC&oi=fnd&pg=PP1&dq=How%2Bdoes%2Bthe%2Bprogram%2Bversion%2Bin%2Bthe%2Bprivate%2Bworkspace%2Bdiffer%2Bfrom%2Bthe%2Bbaseline%2Bversion%3F%2BWhat%2Bis%2Ba%2Bconflict%2Bin%2Bterms%2Bof%2Btwo%2Bdifferent%2Bupdates%2Bto%2Ba%2Bfile%3F%2BHow%2Bdoes%2Bit%2Bget%2Bresolved%3F&ots=59jhxdrWPR&sig=7eFjQOtfos3UOeiQIkF0x5ImcSM#v=onepage&q&f=false

Question Eight

What is the build and what is the result of the build?

In software engineering, the "build" refers to the process of converting source code files into standalone software artifact(s) that can be run on a computer, or the outcome of this process. This involves compiling the code, linking it with libraries, and packaging it into executable programs or other types of deliverables, such as libraries or web applications. The "result of the build" is the executable application or output produced after the build process has been completed successfully. This result can be a binary, a library, a web application package, or any other type of software component that is ready for deployment or distribution. The build process is often automated using tools like Make, Maven, Gradle, or continuous integration systems, ensuring consistency and efficiency in producing software products.

Hoste, K., Timmerman, J., Georges, A., & De Weirdt, S. (2012). EasyBuild: Building software with ease. In 2012 SC Companion: High Performance Computing, Networking Storage and Analysis (pp. 572-582). Salt Lake City, UT, USA.

Question Nine

What is the three-tier architecture?

The three-tier architecture is a widely adopted design framework in software engineering for developing web applications and information systems. It logically separates an application into three distinct layers: the presentation layer, the application (or business logic) layer, and the data layer. The presentation layer is

responsible for the user interface and user experience, interacting directly with users. The application layer contains the business logic, processing user requests, performing operations, and making decisions. Finally, the data layer manages data storage and retrieval, interfacing with databases or other data sources. This architecture promotes separation of concerns, making applications more manageable, scalable, and maintainable by isolating the user interface, business logic, and data storage functions.

Abba Ari, A. A., Djedouboum, A. C., Gueroui, A. M., Thiare, O., Mohamadou, A., & Aliouat, Z. (2020, August 4). A three-tier architecture of large-scale wireless sensor networks for Big Data Collection. MDPI. https://www.mdpi.com/2076-3417/10/15/5382

Question Ten

What is polymorphism in technology? Give an example.

Polymorphism, in the context of technology and more specifically in object-oriented programming, refers to the ability of different objects to respond to the same function call in different ways. It allows methods to do different things based on the object that is invoking them. One common example of polymorphism is the shape class hierarchy. Suppose there is a base class named Shape with a method called draw(). Different subclasses such as Circle, Square, and Triangle might implement the draw() method in different ways to draw the specific shape they represent. When a program calls the draw() method on a Shape object, the actual method that gets executed depends on the kind of object it is, whether it's a Circle, Square, or Triangle. This allows for code that can work with objects of different types through the same interface, enhancing flexibility and reusability

LaFramboise, T. (2009). Single nucleotide polymorphism arrays: a decade of biological, computational and technological advances. Nucleic Acids Research, 37(13), 4181-4193.

NOTE:

```
➜  10-31-23 cat person_Gen.py
import random
def Gen():
    """ Creates and returns a generated name """
    first=["John","Paul","Goerge","Ringo"]
    last1=["Br","Cl","P","W","Wr","Wren","Wron"]
    last2=["ee","ie","a","i","kow","e","er"]
    last3="bcdfghjklmnpqrstuvwxyz"
    gen_name=str(random.choice(last1))+random.choice(last2)+random.choice(last3)+" " + random.choice(first)
    skills=["call-tree","IT-Generalist","IT-Security","PHY-Security","Welder","Tool-and-Die","Standup-Comedian"]
    location = ["East","West","North","South"]
    employee={}
    domain=['aa','b','bn','bc','de','q']
    employee[str(random.randrange(1000))+random.choice(domain)]=gen_name+" "+random.choice(skills)+" "+random.choice(location)
    return employee
print(Gen())
print(Gen.__doc__)
```