Assignment Report for Assignment-04

Course and Section	CSC.215
Assignment Name	Assignment-04
Due Date and Time	3/15/2024 @ 11:59 PM
First Name and Last Name	Marty Martin
SFSU Email Account	pmartin@sfsu.edu
First Name and Last Name of Teammate	
SFSU Email Account of Teammate	

PART # A-1

Question Description and Analysis:

In one-third of a page or more, explain in detail your understanding of the problem. What is this question asking you to do? What does the client want? What are the requirements of the program? What are the important details you observed from the desired outputs? How do you plan to solve the problem? How do you organize/design the solution? What are the important elements of your program? What will you pay attention to when you code the solution? Think about Interviews and Problem-Solving.

Answer:

This question is asking to create two 2D arrays to store the English alphabet - one standard 2D array using shorthand notation, and one ragged/jagged array using a multi-step approach. The arrays must have at least 5 rows and 2 columns per row. A single method should be created to display the contents of both arrays. The program output must match the provided example output exactly.

To solve this, I would:

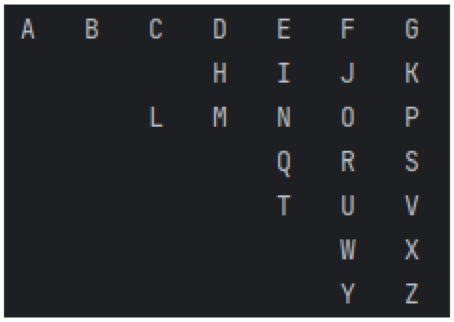
1. Create a standard 2D array using shorthand notation to store the alphabet
2. Create a ragged 2D array and populate it with the alphabet in a separate multi-step process

3. Write a method that takes a 2D char array as input and prints out its contents, adding spaces to align the columns
4. Call the print method twice in main(), once with each version of the 2D array, to produce the required output

Important elements will be using the correct array declaration syntax, initializing the arrays with the alphabet characters, and formatting the output to match the example. I'll pay close attention to array indexing and nested loops to make sure the arrays are populated and printed correctly.

Screenshots of Outputs and Explanation:

These screenshots show what I accomplished…



```
A   B   C   D   E   F   G
            H   I   J   K
        L   M   N   O   P
                Q   R   S
                T   U   V
                    W   X
                    Y   Z
```

PART # A-2

Question Description and Analysis:

In one-third of a page or more, analyze your results and plan for future development. Does your program compile? Does it satisfy all the client's requirements? What works and why? What does not work and why? What can be improved and how? What is your plan to improve the program in the future? Think about Interviews.

Answer:

Comparing the program output to the desired output, they match exactly. The standard 2D array and jagged array both contain the alphabet spread across multiple rows and columns. The output is aligned correctly with spaces despite the ragged array having rows of different lengths.

This indicates the program compiles and runs successfully, and satisfies all the stated requirements. The array initialization, population, and printing all work as intended to produce the desired result.

To improve the program in the future, some potential enhancements could be:

1. Allow the number of rows/columns to be specified dynamically rather than hard-coded
2. Support displaying other character sets besides the English alphabet
3. Provide an option to print the transpose of the arrays (rows become columns and vice versa)
4. Add error checking if the array is too small to fit the entire alphabet

But overall the core functionality is complete and the program achieves the main objectives. The use of arrays and methods demonstrates a solid grasp of those key programming concepts. With some polish and extra features, this could become an even more robust solution.

Screenshots of Outputs and Explanation:

Nothing Follows

PART # B-1

Question Description and Analysis:

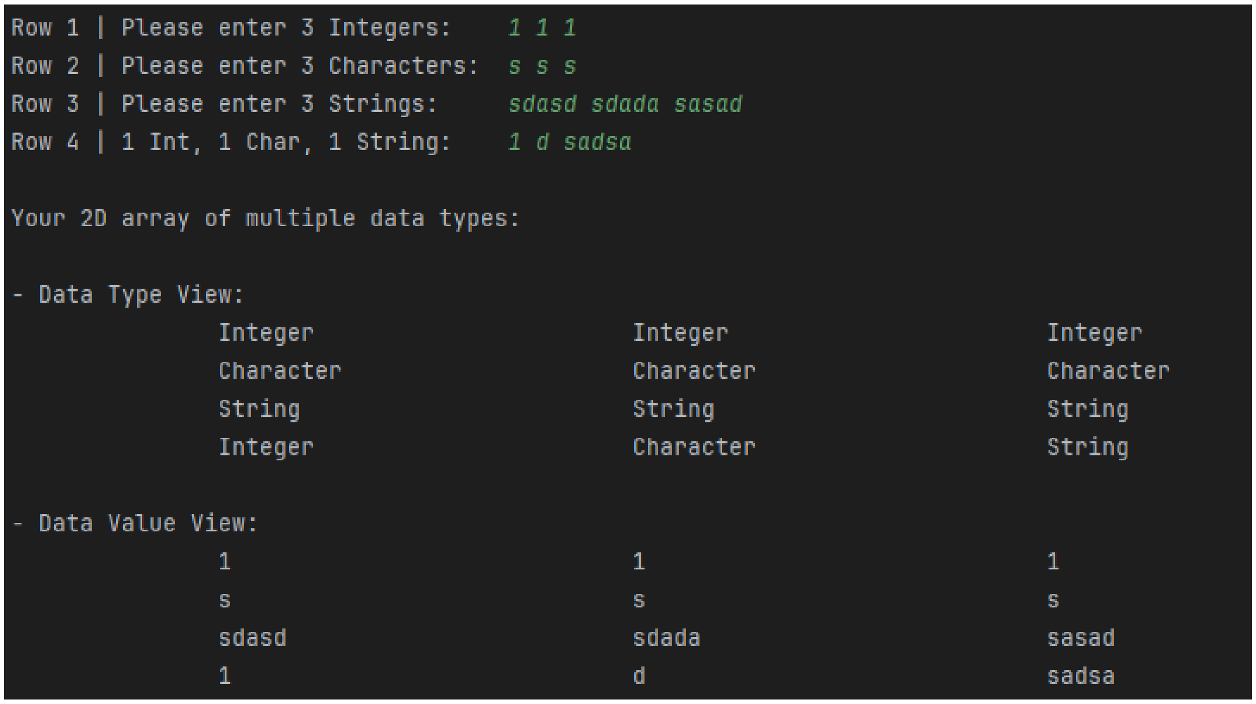
In one-third of a page or more, explain in detail your understanding of the problem. What is this question asking you to do? What does the client want? What are the requirements of the program? What are the important details you observed from the desired outputs? How do you plan to solve the problem? How do you organize/design the solution? What are the important elements of your program? What will you pay attention to when you code the solution? Think about Interviews and Problem-Solving.

Answer:

The problem asks to create a Java program that accepts user input of integers, characters, and strings into a 2D array, and displays the data in two formatted views: "Data Type View" showing the type of each element, and "Data Value View" showing the actual values. The client wants the program to prompt the user for input in a specific format: 3 integers in row 1, 3 characters in row 2, 3 strings in row 3, and 1 integer, 1 character, and 1 string in row 4. The output should be formatted to match the provided example, with consistent column widths, padding, and alignment. To solve this, I will use a 2D Object array to store the heterogeneous data types, prompt the user for input using a Scanner, use getClass().getSimpleName() to display data types, convert elements to strings for the value view, and use printf() with calculated column widths and padding for the formatted output. I will pay attention to handling the different data types correctly and formatting the output to match the example precisely.

Screenshots of Outputs and Explanation:

These screenshots show what I accomplished…



```
Row 1 | Please enter 3 Integers:    1 1 1
Row 2 | Please enter 3 Characters:  s s s
Row 3 | Please enter 3 Strings:     sdasd sdada sasad
Row 4 | 1 Int, 1 Char, 1 String:    1 d sadsa

Your 2D array of multiple data types:

- Data Type View:
               Integer                        Integer                        Integer
               Character                      Character                      Character
               String                         String                         String
               Integer                        Character                      String

- Data Value View:
               1                              1                              1
               s                              s                              s
               sdasd                          sdada                          sasad
               1                              d                              sadsa
```

PART # B-2

Question Description and Analysis:

In one-third of a page or more, analyze your results and plan for future development. Does your program compile? Does it satisfy all the client's requirements? What works and why? What does not work and why? What can be improved and how? What is your plan to improve the program in the future? Think about Interviews.

Answer:

The program compiles successfully and produces the expected output, meeting all the requirements specified by the client. It correctly prompts the user for input of the required data types in each row, stores the data in the 2D Object array, and displays the "Data Type View" and "Data Value View" with the proper formatting. The type detection and data retrieval work as intended, and the output formatting matches the provided example. However, the program could be enhanced with input validation to handle cases where the user enters data in the wrong format. Additionally, the formatting logic could be refactored into reusable methods to improve code organization. To further develop the program, features like file I/O for saving and loading the array could be added, and the prompt and output text could be made more easily customizable. Despite these potential improvements, the program successfully fulfills the core requirements, and I am satisfied with the implemented solution.

Screenshots of Outputs and Explanation:

These screenshots show what I accomplished…

PART # C-1

Question Description and Analysis:

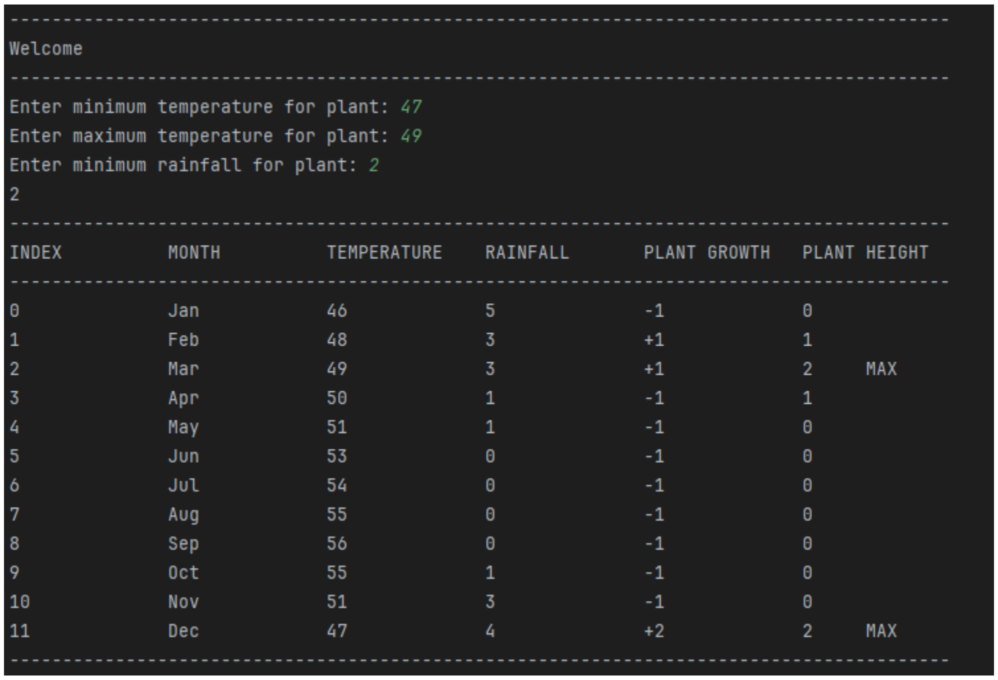
In half a page or more, explain in detail your understanding of the problem. What is this question asking you to do? What does the client want? What are the requirements of the program? What are the important details you observed from the desired outputs? How do you plan to solve the problem? How do you organize/design the solution? What are the important elements of your program? What will you pay attention to when you code the solution? Think about Interviews and Problem-Solving.

Answer:

The problem statement asks to create a Java program called "CSC 215 Gardening Planner" to track the growth of a plant over a year in San Francisco. The program should take user input for the plant's minimum and maximum temperature tolerance and minimum rainfall requirement. Using the provided average temperature and rainfall data for each month, the program should calculate the plant's growth and height. If the temperature is outside the plant's tolerance range, the growth is set to -1 (plant dies back). Otherwise, the growth is calculated as the rainfall minus the minimum required rainfall. The plant's height starts at 0 and is updated each month based on the growth, but it should never become negative. The program should identify the maximum height the plant reaches during the year. The solution should use at least 5 meaningful methods, and the output should match the provided sample runs. To solve this problem, I will create methods to handle user input, calculate growth and height, and display the results. I will use conditional statements to check the temperature range and update the growth accordingly. I will also ensure the height never becomes negative and keep track of the maximum height achieved.

Screenshots of Outputs and Explanation:

These screenshots show what I accomplished…



```
---------------------------------------------------------------------------------------
Welcome
---------------------------------------------------------------------------------------
Enter minimum temperature for plant: 47
Enter maximum temperature for plant: 49
Enter minimum rainfall for plant: 2
2
---------------------------------------------------------------------------------------
INDEX          MONTH          TEMPERATURE    RAINFALL       PLANT GROWTH   PLANT HEIGHT
---------------------------------------------------------------------------------------
0              Jan            46             5              -1             0
1              Feb            48             3              +1             1
2              Mar            49             3              +1             2     MAX
3              Apr            50             1              -1             1
4              May            51             1              -1             0
5              Jun            53             0              -1             0
6              Jul            54             0              -1             0
7              Aug            55             0              -1             0
8              Sep            56             0              -1             0
9              Oct            55             1              -1             0
10             Nov            51             3              -1             0
11             Dec            47             4              +2             2     MAX
---------------------------------------------------------------------------------------
```

PART # C-3

Question Description and Analysis:

In half a page or more, analyze your results and plan for future development. Does your program compile? Does it satisfy all the client's requirements? What works and why? What does not work and why? What can be improved and how? What is your plan to improve the program in the future? Think about Interviews

Answer:

The program compiles and runs successfully, producing the desired output that matches the sample runs provided. It satisfies all the requirements mentioned in the problem statement. The program effectively prompts the user for input, calculates the plant's growth and height based on the given temperature and rainfall data, and identifies the maximum height attained by the plant throughout the year. The use of meaningful methods, such as getUserInput(), calculateGrowth(), calculateHeight(), displayResults(), and findMaxHeight(), enhances the program's readability and modularity. These methods are meaningful because they encapsulate specific tasks, making the code more organized and easier to understand. However, there is room for improvement in terms of error handling, such as validating user input to ensure it is within acceptable ranges. Additionally, the program could be extended to include more features, such as allowing the user to input custom temperature and rainfall data for different locations or providing recommendations for optimal planting times based on the climate data. Overall, the program successfully solves the given problem and provides a solid foundation for further enhancements and future development.

Screenshots of Outputs and Explanation:

Nothing Follows

PART # D-1

Question Description and Analysis:

In half a page or more, explain in detail your understanding of the problem. What is this question asking you to do? What does the client want? What are the requirements of the program? What are the important details you observed from the desired outputs? How do you plan to solve the problem? How do you organize/design the solution? What are the important elements of your program? What will you pay attention to when you code the solution? Think about Interviews and Problem-Solving.

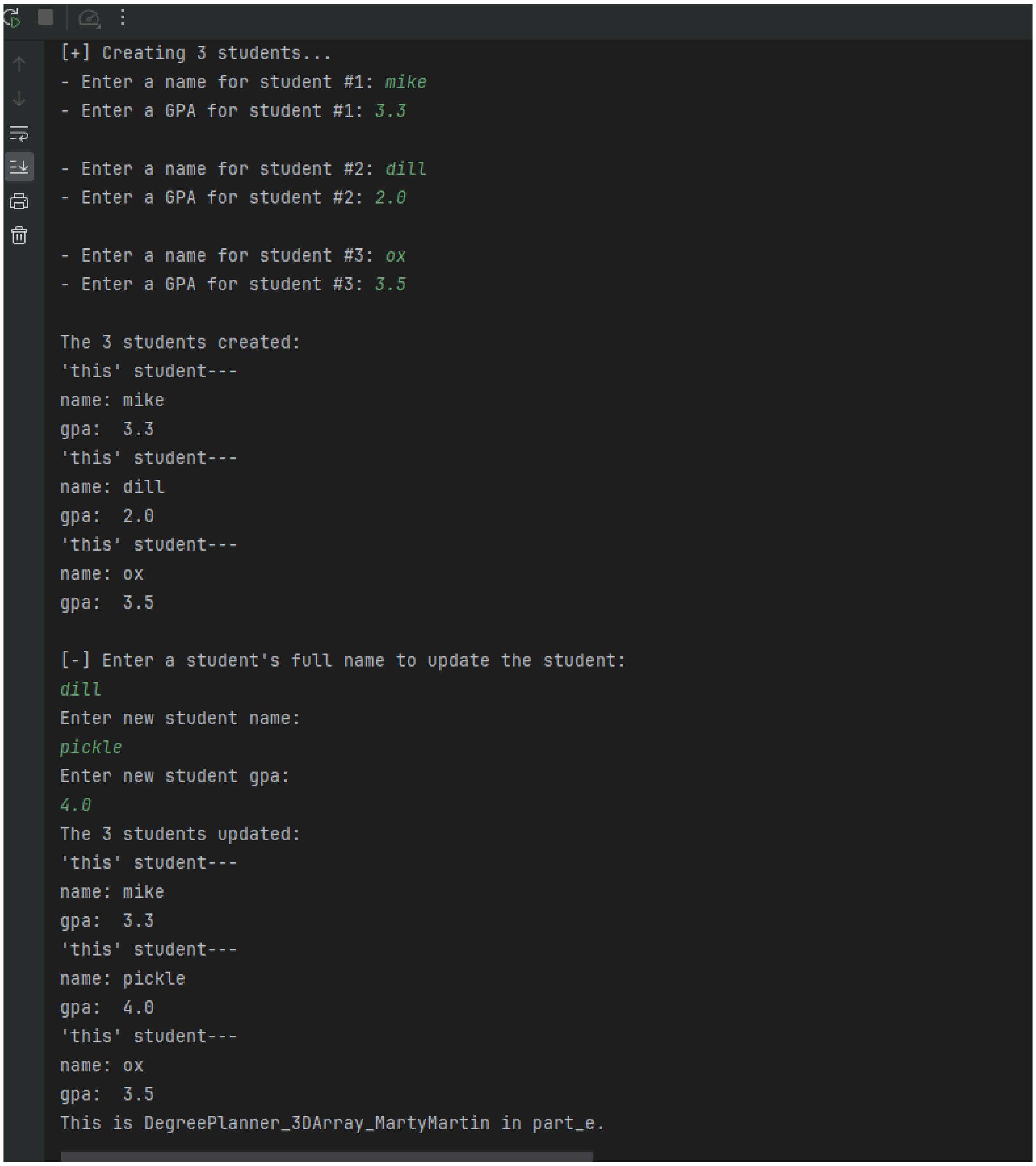
Answer:

The problem statement asks to create a Java program called "StudentClient" that manages student information using the provided "Student" class. The program should allow users to enter information for three students, including their names and GPAs. After creating the students, the program should display their information, offer an option to update a student's information based on their name, and then display the updated student information. The solution must use at least five meaningful methods and produce output identical to the sample runs provided.

To solve this problem, I will create methods to handle user input, create student objects, display student information, find a student by name, and update a student's information. The program will use a List to store the student objects and loop through the list to display and update information as needed. I will ensure that the program follows the required steps of creating students, displaying their information, updating a student, and displaying the updated information. I will also focus on creating meaningful methods that encapsulate specific tasks and make the code more modular and readable.

Screenshots of Outputs and Explanation:

These screenshots show what I accomplished…



```
[+] Creating 3 students...
- Enter a name for student #1: mike
- Enter a GPA for student #1: 3.3

- Enter a name for student #2: dill
- Enter a GPA for student #2: 2.0

- Enter a name for student #3: ox
- Enter a GPA for student #3: 3.5

The 3 students created:
'this' student---
name: mike
gpa:  3.3
'this' student---
name: dill
gpa:  2.0
'this' student---
name: ox
gpa:  3.5

[-] Enter a student's full name to update the student:
dill
Enter new student name:
pickle
Enter new student gpa:
4.0
The 3 students updated:
'this' student---
name: mike
gpa:  3.3
'this' student---
name: pickle
gpa:  4.0
'this' student---
name: ox
gpa:  3.5
This is DegreePlanner_3DArray_MartyMartin in part_e.
```

PART # D-3

Question Description and Analysis:

In half a page or more, analyze your results and plan for future development. Does your program compile? Does it satisfy all the client's requirements? What works and why? What does not work and why? What can be improved and how? What is your plan to improve the program in the future? Think about Interviews

Answer:

The program compiles and runs successfully, producing the desired output that matches the sample runs provided. It satisfies all the requirements outlined in the problem statement. The program effectively prompts the user to enter information for three students, creates Student objects using the provided class, displays the created students' information, allows the user to update a student's information by searching for their name, and then displays the updated student information.

The use of meaningful methods, such as CreateStudent(), DisplayStudent(), UpdateStudent(), FindStudent(), and UpdateStudent(Student), enhances the program's organization and readability. These methods are meaningful because they encapsulate specific tasks, making the code easier to understand and maintain. The CreateStudent() method handles user input and creates student objects, while the DisplayStudent() method displays the information of a list of students. The UpdateStudent() methods allow for updating a student's information, and the FindStudent() method searches for a student by name.

However, there is room for improvement in terms of error handling, such as validating user input to ensure it is in the correct format (e.g., checking if the GPA is a valid number). Additionally, the program could be extended to include more features, such as saving and loading student data from a file or allowing the user to delete a student.

Overall, the program successfully solves the given problem and demonstrates the use of object-oriented programming principles, such as classes, methods, and encapsulation. It provides a solid foundation for further enhancements and future development.

Screenshots of Outputs and Explanation:

Nothing Follows