**Program Specifications – Student Data Analysis**

Soumil Vavikar

Purdue University Global

IN501 – Fundamentals of Computer Programming

Dr. Mortoza Abdullah

January 31, 2024

**Program Specifications – Student Data Analysis**

This document contains the IO chart and pseudocode for the project that has to be completed as part of the curriculum for the course “IN501 – Fundamentals of Computer Programming”. Per the project specification document, the program to be developed/written will be used to analyze students' data seeking either of the two courses - MSIT and MSCM. The program should be able to take user input for which operation needs to be performed, then do necessary calculations and display results in the expected format.

**Input Processing Output (IPO) Chart**

|  |  |  |
| --- | --- | --- |
| Input | Processing | Output |
| * User input 1: “Display average grade for all students.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Calculate average grades for all students. 4. Print the average grades for all the students to the console. | 1. All students' average grades are printed in the console. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 2: “Display average grade for each program.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Separate student's lists into two lists, one for students pursuing MSIT and the other for students pursuing MSCM. 4. Calculate average grades for students in MSIT and MSCM. 5. Print the average grades for degree courses – MSIT and MSCM to the console. | 1. The average grades for students pursuing MSIT and MSCM are printed in the console. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 3: “Display highest grade record.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Iterate through the list of students and find the record with the highest grade. 4. Print the record to the console. | 1. The record/student details with the highest grade are printed in the console. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 4: “Display lowest grade record.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Iterate through the list of students and find the record with the lowest grade. 4. Print the record to the console. | 1. The record/student details with the lowest grade are printed in the console. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 5: “Display students in MSIT.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Iterate through the list of students and collect the students pursuing degree courses MSIT in a list. 4. Print the list to the console. | 1. The list of students pursuing a degree in MSIT is printed in the console. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 6: “Display students in MSCM.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Iterate through the list of students and collect the students pursuing degree courses MSCM in a list. 4. Print the list to the console. | 1. The list of students pursuing a degree in MSCM is printed in the console. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 7: “Display all students in sorted order by student ID.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Validate each record, convert each valid record into a student object, and put them in a student’s list. 3. Sort the students' list based on the student\_id field in each student object within the list. 4. Print the sorted list on the console. | 1. The list of students is printed in the console in the sorted order by student\_id field. 2. Print the operations the user can perform. 3. Ask the user to input a valid option to perform the next operation. |
| * User input 8: “Display invalid records.” * Default Input – STUDENTDATA.TXT | 1. Read the STUDENTDATA.TXT file line by line and treat each line as one student record. 2. Perform validations and checks on each record. 3. Add the record/line to the invalid\_students list if any validations fail. Find the list of validations in the project specifications document. 4. If the invalid\_students list is not empty:    1. Print the invalid records on the console.    2. Write the invalid records to the “BADRECORDS.TXT.”    3. Print “BADRECORDS.TXT has been created.” to the console. | 1. Invalid student records are printed on the console. 2. “BADRECORDS.TXT” is generated. 3. “BADRECORDS.TXT has been created.” It is printed on the console. |
| User input 9: “Exit Program.” | Print “Exiting the Program.” Furthermore, exit the program. | Console output: “Exiting the Program.” |
| User inputs any number or string other than a number ranging between 1 to 9 | 1. Validate the user input. 2. If the user input is not an integer between 1 and 9, print the error message: “Invalid menu option selected. Valid values are between 1 and 9. Please try again.” 3. Print the options and ask the user to input one between 1 and 9. | 1. Error message in the console: “Invalid menu option selected. Valid values are between 1 and 9. Please try again.” 2. Request the user to input again. |
| Either the STUDENTDATA.TXT is provided with no valid records, or it is not present. | 1. Check if the STUDENTDATA.TXT is present and has at least one valid record. 2. If the file is not provided or no valid records are present, print an error message on the console: “Exception occurred while reading the input file.” 3. Exit the program. | 1. An error message is printed on the console: “Exception occurred while reading the input file.” 2. The program is exited. |

**Pseudocode**

<TBD>