|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Coventry University/CityU**  **This assignment (hard copy and also in electronic form to Canvas ) is to be submitted to the Front Office of SCOPE in 2410, 2/F, Li Dak Sum Yip Yio Chin Academic Building (LI)** | | | | Academic Office Date Stamp | | | |
| **StudentID :**  **Family Name(s):** | |  | | | | | |
| **Forename(s):** | |  | | | | | |
| **Declaration:** I / We declare that this Coursework is my / our own work. Cheating is a serious academic  offence. I / We have read and agree to abide by the University guidance on Academic Honesty. | | | | | | | |
| **Signature(s):**  **Time taken to complete assignment (hrs):** | | | | | | | |
| **Module Code/Title:** CHK122COM Introduction to Algorithms  **Lecturer/Tutor:** Terence Chan  **Hand-out Date:** Week 1  **Assignment No./Title**: Coursework  **Assignment Type**: Individual  Estimated Time (hrs): 30 hours | | | | **Due Date:** Week 9 | | | |
| **Learning Outcomes Assessed (or on separate sheet)** | | | | | | | |
| Intended Module Learning Outcomes 1,2,3,4 | | | | | | | |
| **Assessment Criteria (on separate sheet)** | | | | | | **Max** | **Awarded** |
| (According to the coursework document) | | | | | |  |  |
| **Comments** | | | | | **Total** |  |  |
|  | | | | |  | |  |
| **Assessor’s signature:** | | | **Date:** | | **Final Mark** | |  |
|  |  | | | | | | |

## CHK122COM Introduction to Algorithms

**Assignment**

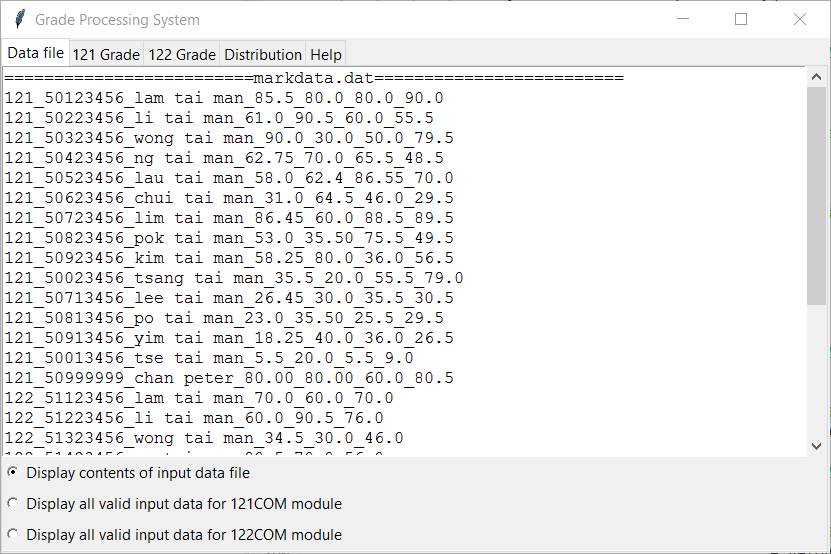
## GUI Grade Processing System

### Description

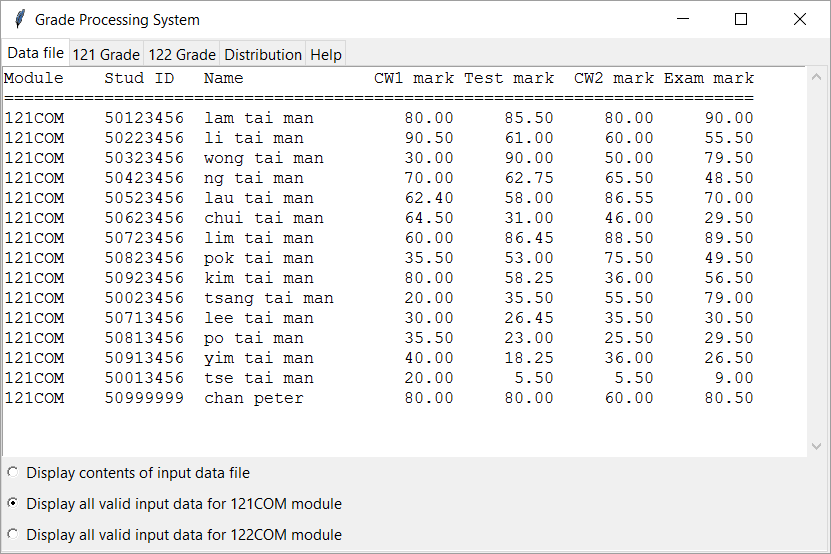
Finish a Python program that allows the user to load grade information from a file of grade data, process grade for different modules, and plot grade distribution using ttk.Notebook widget to display result in separate windows. Your task is to design and implement a driver program GUIgrade.py that uses the classes provided and implements a graphical user interface grade processing system which produces following functions:

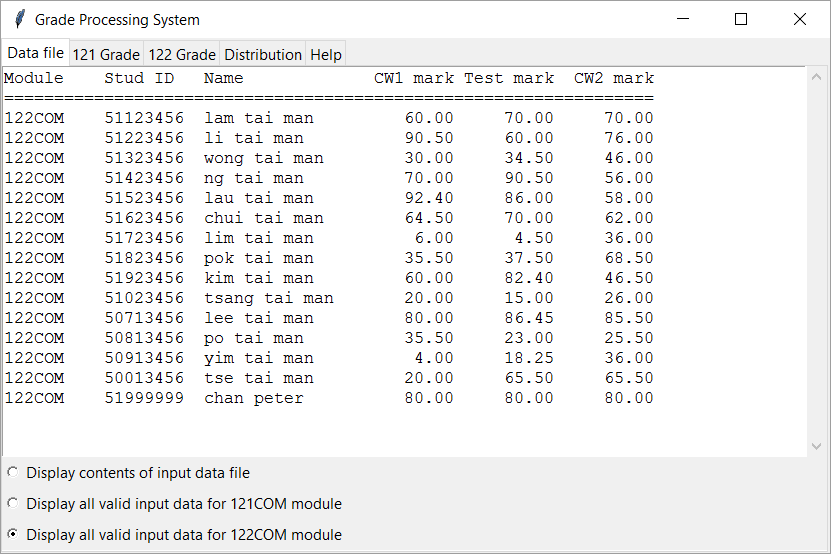
1. **“Data file” window**

The following is a screen shot of the program after a sample data file i.e. markdata.dat has been loaded and displayed.



Once the information is loaded, the mark data will be displayed in a Text component. The user can then selectively view either 1) all grade data, 2) grade data for 121COM, or 3) grade data for 122COM by clicking the Checkbox components.





The student mark data is in the form of a text file markdata.dat.

Sample Student data file “markdata.dat”

121\_50123456\_lam tai man\_85.5\_80.0\_80.0\_90.0

121\_50223456\_li tai man\_61.0\_90.5\_60.0\_55.5

121\_50323456\_wong tai man\_90.0\_30.0\_50.0\_79.5

121\_50423456\_ng tai man\_62.75\_70.0\_65.5\_48.5

121\_50523456\_lau tai man\_58.0\_62.4\_86.55\_70.0

121\_50623456\_chui tai man\_31.0\_64.5\_46.0\_29.5

121\_50723456\_lim tai man\_86.45\_60.0\_88.5\_89.5

121\_50823456\_pok tai man\_53.0\_35.50\_75.5\_49.5

121\_50923456\_kim tai man\_58.25\_80.0\_36.0\_56.5

121\_50023456\_tsang tai man\_35.5\_20.0\_55.5\_79.0

121\_50713456\_lee tai man\_26.45\_30.0\_35.5\_30.5

121\_50813456\_po tai man\_23.0\_35.50\_25.5\_29.5

121\_50913456\_yim tai man\_18.25\_40.0\_36.0\_26.5

121\_50013456\_tse tai man\_5.5\_20.0\_5.5\_9.0

121\_50999999\_chan peter\_80.00\_80.00\_60.0\_80.5

122\_51123456\_lam tai man\_70.0\_60.0\_70.0

122\_51223456\_li tai man\_60.0\_90.5\_76.0

122\_51323456\_wong tai man\_34.5\_30.0\_46.0

122\_51423456\_ng tai man\_90.5\_70.0\_56.0

122\_51523456\_lau tai man\_86.0\_92.4\_58.0

122\_51623456\_chui tai man\_70.0\_64.5\_62.0

122\_51723456\_lim tai man\_4.5\_6.0\_36.0

122\_51823456\_pok tai man\_37.5\_35.50\_68.5

122\_51923456\_kim tai man\_82.4\_60.0\_46.5

122\_51023456\_tsang tai man\_15.0\_20.0\_26.0

122\_50713456\_lee tai man\_86.45\_80.0\_85.5

122\_50813456\_po tai man\_23.0\_35.50\_25.5

122\_50913456\_yim tai man\_18.25\_4.0\_36.0

122\_50013456\_tse tai man\_65.5\_20.0\_65.5

122\_51999999\_chan peter\_80.00\_80.00\_80.0

It contains one line of information for each student represented. Each line contains the following information, in the order shown.

Breakdown Of A 121COM Student Entry

1. an int - the module code
2. an int - the student ID number
3. a String - the name of the student
4. a float - the coursework1 mark
5. a float - the test mark
6. a float - the coursework2 mark
7. a float - the examination mark

The items on each line are delimited by the underscore character ("\_") (since the student name may contain spaces, the space character could not be used as the delimiter). A sample entry line follows.

Sample 121COM Student Entry

121\_50123456\_lam tai man\_85.5\_80.0\_80.0\_90.0

Breakdown of A 122COM Student Entry

1. an int - the module code
2. an int - the student ID number
3. a String - the name of the student
4. a float - the coursework1 mark
5. a float - the test mark
6. a float - the coursework2 mark

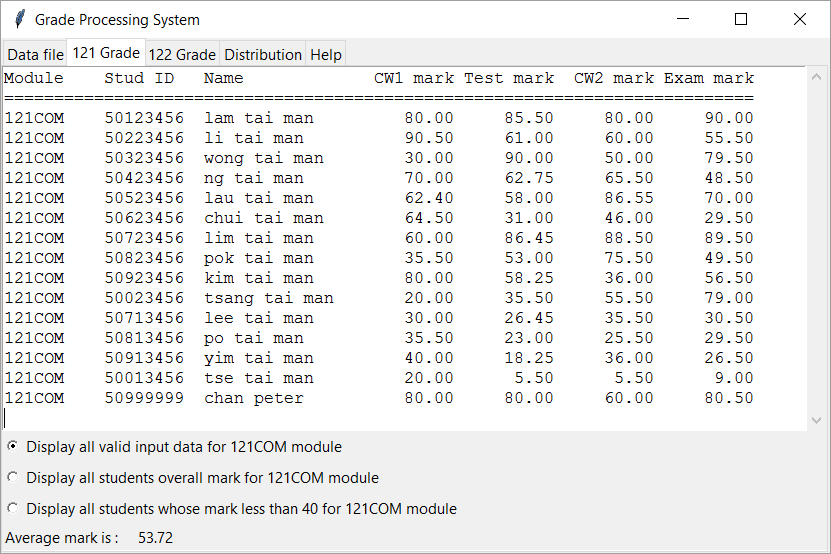
The items on each line are delimited by the underscore character ("\_") (since the student name may contain spaces, the space character could not be used as the delimiter). A sample entry line follows.

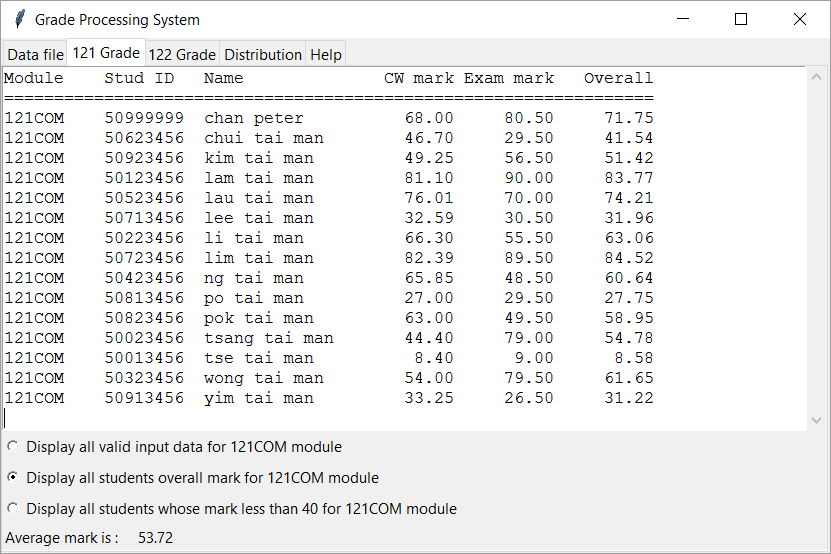
Sample 122COM Student Entry

122\_51123456\_lam tai man\_70.0\_60.0\_70.0

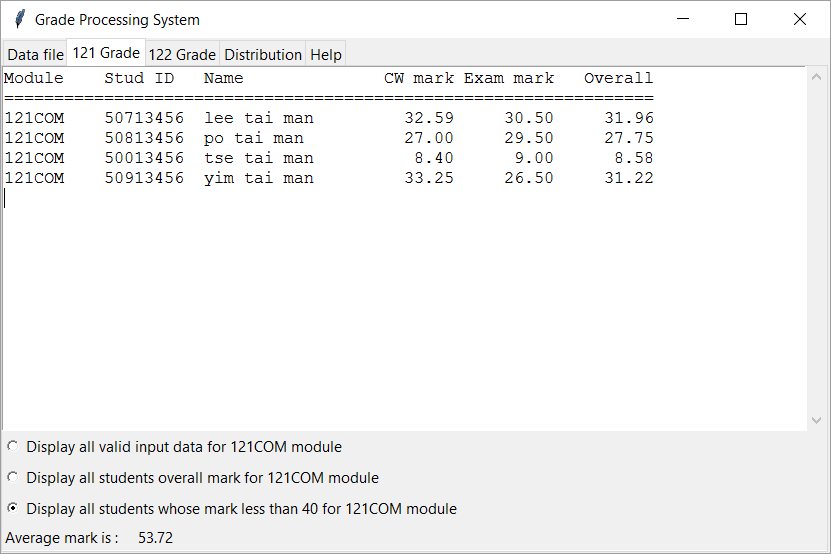
1. **“121/122 Grade” window**

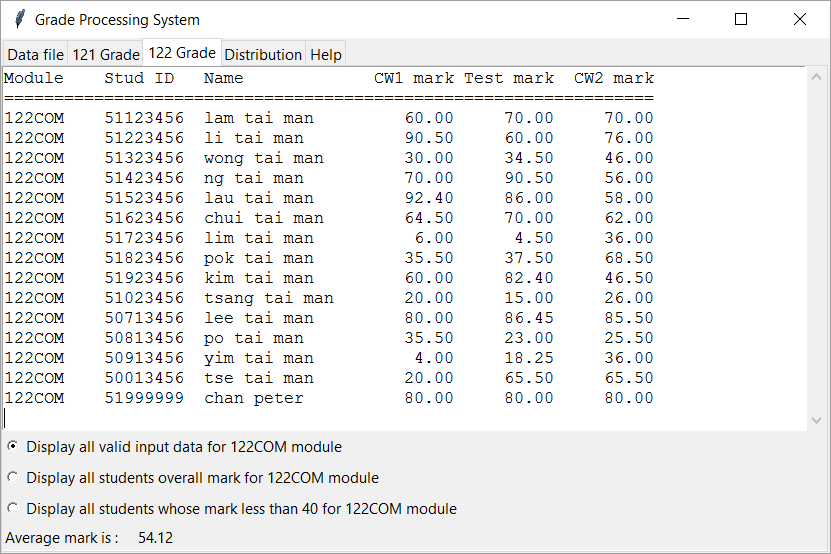
When “**121/122 Grade”** tab is selected to display overall grade of 121COM/122COM students. The user can then selectively view either 1) all grade data, 2) all students’ overall marks, or 3) students whose overall marks less than 40 by clicking the Checkbox components. The class average mark is also displayed in the user interface.

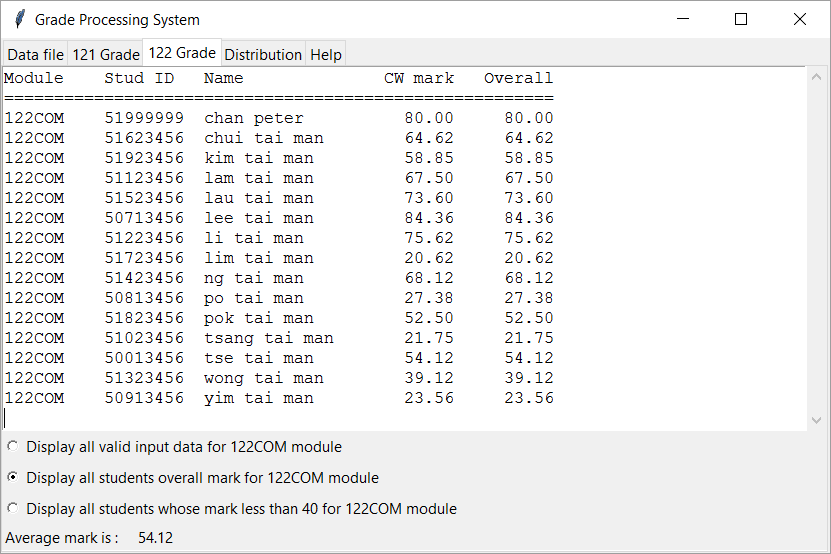


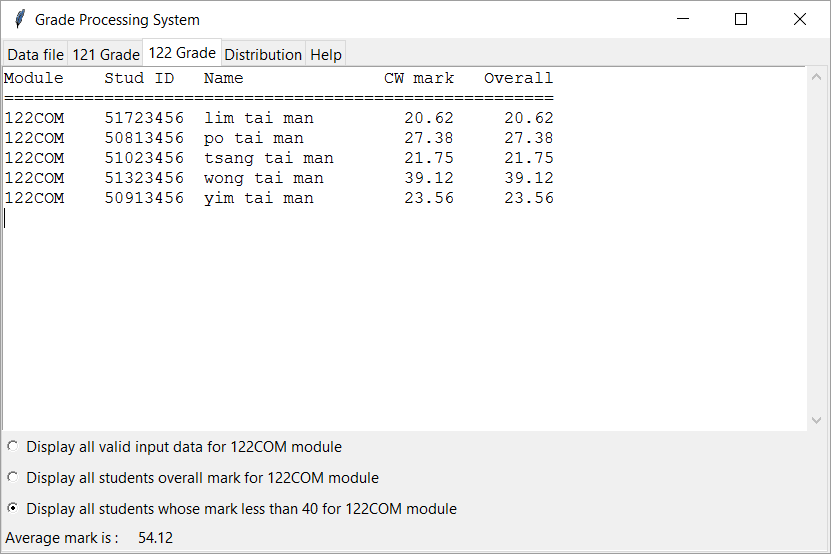
When Second button is selected to display overall grade of students. 

When third button is selected to display all failure cases.





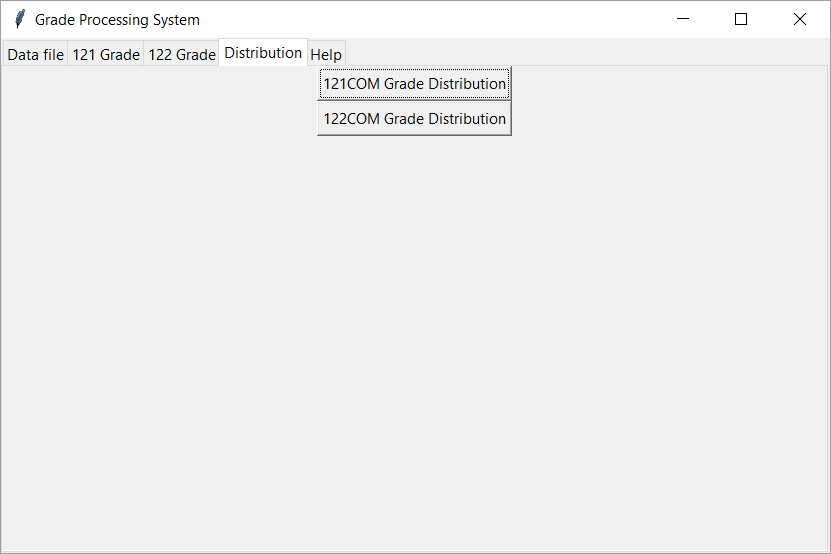


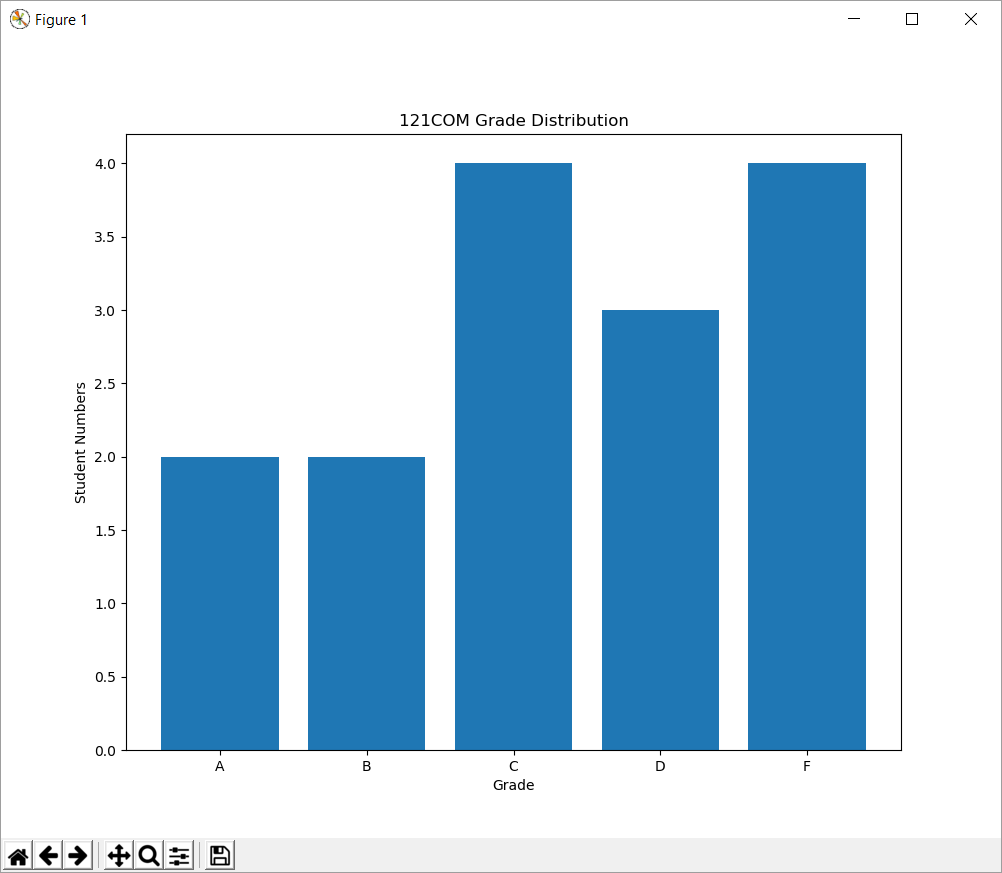


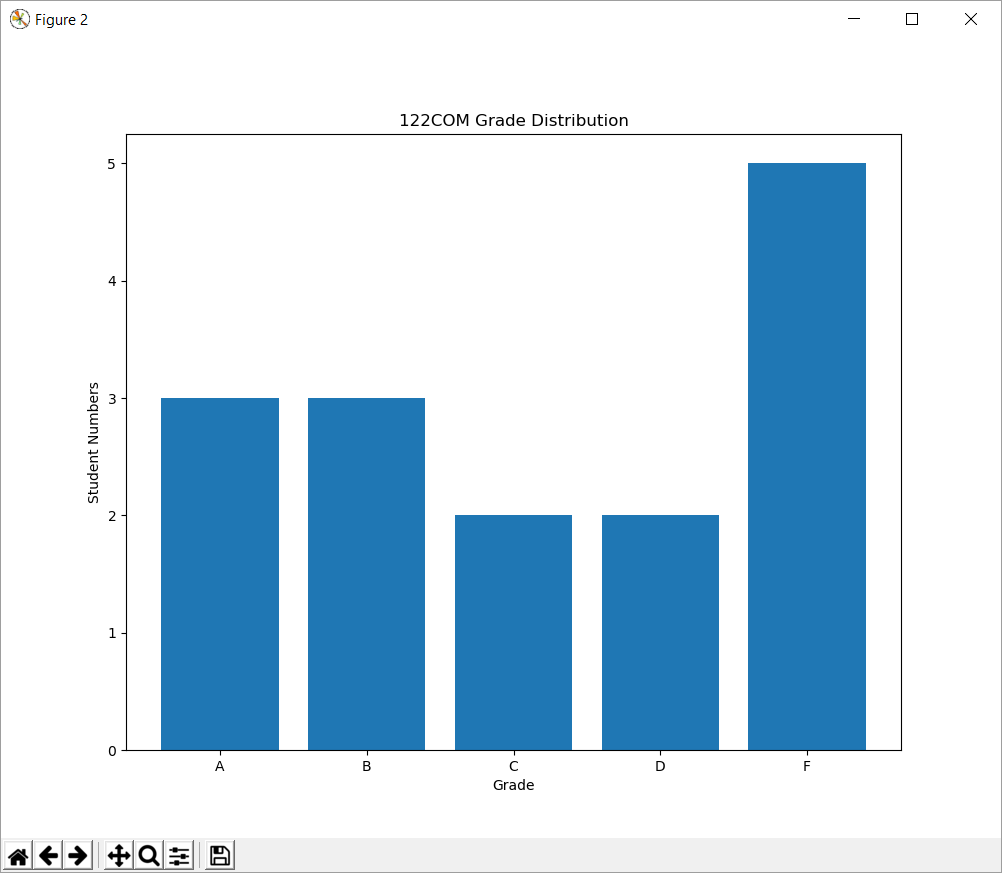
1. **“Distribution” window**

When “**Distrbution window”** tab is selected, two Button components will be displayed. The user can then selectively plot distribution of grade for 121COM/122COM module using grade boundary as follows:

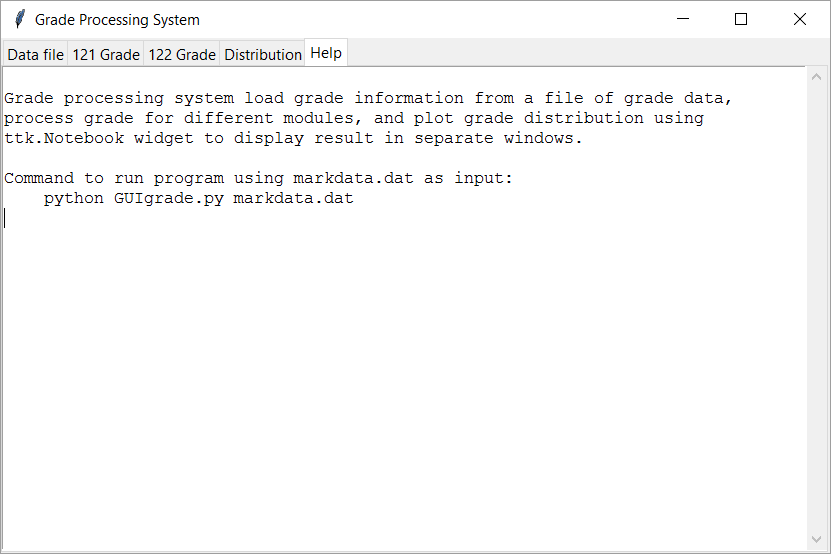
|  |  |
| --- | --- |
| **Scores range** | **Grade** |
| 0 – 39.99 | F |
| 40 – 49.99 | D |
| 50 – 64.99 | C |
| 65 – 74.99 | B |
| 75 – 100 | A |







**iv. Finally, “Help” window will display brief description about the program.**



In GUIgrade, the class that defines the graphical user interface and extends tkinter.Frame. The following components must be used:

* ttk.Notebook widget to display result in separate windows
* Scrolledtext components to display the input and grade data
* Three Radiobutton components, when selected, fills the Scrolledtext components with output
* Label component which displays the class average mark
* Button components, when click, plot bar chart for grade distribution

Use exception handling to catch errors you find or anticipate. Test your program with the following test case:

* Test case 1: sample file “markdata.dat”
* Test case 2: empty file
* Test case 3: missing fields
* Test case 4: invalid fields
* Test case 5: repeated student record

**Files**

Download coursework.zip and extract the archive that contains the following.

* IndividualAssignment.docx – The assignment specification
* Student.py - The complete implementation of the class Student
* Student121.py - The partly complete implementation of the class Student121
* Student122.py - The partly complete implementation of the class Student122
* GUIgrade.py - A simple template with header, import statements, and first window to display a data file. i.e. *python GUIgrade.py markdata.dat will display contents of markdata.dat*
* markdata.dat - A sample input file.
* markdata1.dat - A sample input file.
* markdata2.dat - A sample input file.

### What You Are To Do

This assignment consists of three parts: class Student121, class Student122, and the graphical user interface grade processing program (class GUIgrade). First, complete the constructor in Student121 and Student122. To handle errors more efficiently, the constructor accepts one String arguments instead of numbers of data elements. “studentData” is a String expected to contain a line of student data from mark data file. This String must be parsed into different pieces of information expected to exist in that line. Each piece must be appropriately stored in the corresponding Student instance variable. If the parsing fails, the constructor should throw ValueError Exception, providing type of error e.g. "invalid mark value" or "invalid student name" etc. as an argument to the exception. The parsing should fail if studentData does not contain the exact number of items and each items of the expected format. For example, if a float was expected from the current token, and the token could not be parsed to a float, ValueError should be thrown and an error message should be display in the console e.g.

“could not convert string to float: '80a.0' in input 121\_50123456\_lam tai man\_85.5\_80a.0\_80.0\_90.0”

will be printed when “121\_50123456\_lam tai man\_85.5\_80a.0\_80.0\_90.0” is processed.

To efficiently complete this assignment, you are encouraged to code and test each class and window before moving on to the next.

Upon completion, submit **only** the following.

1. GUIgrade.py, Student121.py, and Student122.py
2. Test data files used
3. Documentation of your program and testing.

**Assessment criteria:**

Assessment criteria:

|  |  |
| --- | --- |
| **Activity/Task** | **Assessment weight** |
| Problem Specification and Design:   * identify input requirements in term of value range, format, restrictions/tolerance to input, using external or internal data, and way to terminate data input; identify output requirements in term of values produced, format, and amount of output. [5 %] * identify special relationship between input and output (for example, how the program handles invalid mark value), and error handling/recovery procedure. [5 %] * object model of the your system (showing data fields and methods) [5%] | 15% |
| Python Implementation   * you should follow the [Style Guide for Python Code](https://www.python.org/dev/peps/pep-0008/) in writting your program. Internal program documentation is required and should include the program/module header and usage of descriptive names and comments. (10%) * implementation of constructor in students class. (5%) * use of all components required in GUIgrade class. (10%) * produce the output required. (25%) | 50% |
| Program correctness   * test the program with valid case data, boundary case data, special case data, and invalid case data * test report of all cases required | 30% |
| User and technical documentation   * descriptions on commands to invoke the application, details of interactions, explanation of messages produced, and limitations of the application. | 5% |