Design Report

Kennard Wang 1809853Z-I011-0045 2019.11.6

1. Distribution of Task:

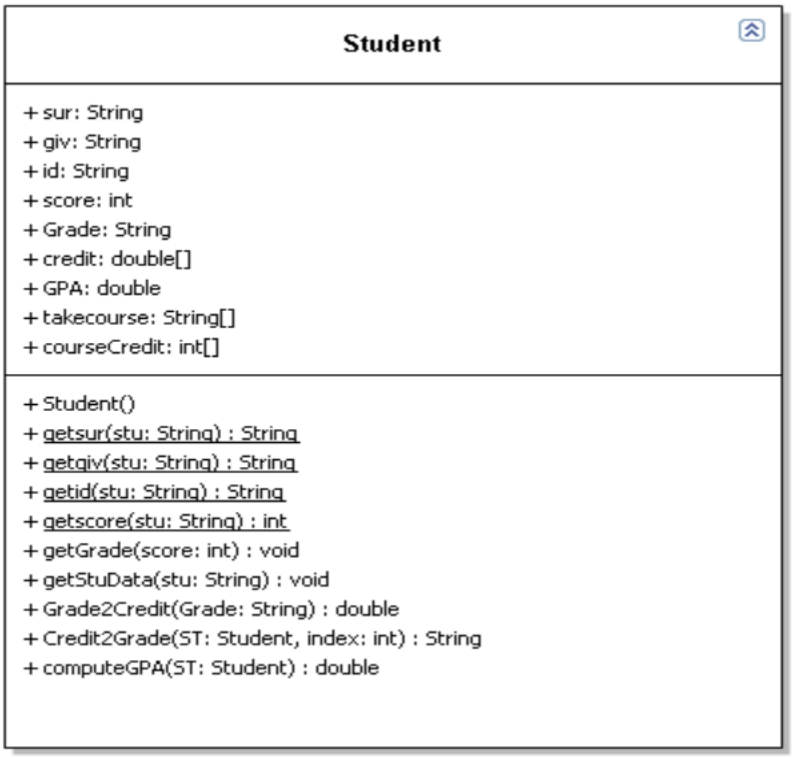
The UGMS(GUI) program is made by Mr. Wang Yuyang (Kennard Wang), who owns the copyright and the right of explanation. The latest version has been updated to 2.1.

2. System Design:

The UGMS(GUI) has a complex class diagram, which will be illustrated later. Also the more detailed description of each class member will be showed in "Data Structure" part.

Student Class

```
Description:
This file contains the Student class, which stores data of all
students after inputting a course file.
```

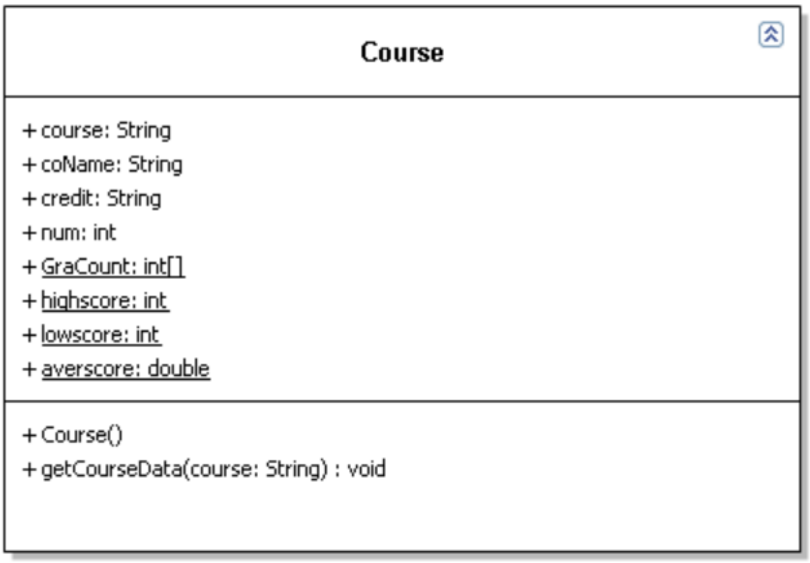


```
Student

+sur: String
+giv: String
+id: String
+score: int
+Grade: String
+credit: double[]
+GPA: double
+takecourse: String[]
+courseCredit: int[]

+Student()
+getsur(stu: String) : String
+getgiv(stu: String) : String
+getid(stu: String) : String
+getscore(stu: String) : int
+getGrade(score: int) : void
+getStuData(stu: String) : void
+Grade2Credit(Grade: String) : double
+Credit2Grade(ST: Student, index: int) : String
+computeGPA(ST: Student) : double
```

Course Class

Description:

This file contains the Course class, which stores data of each input course.



Course

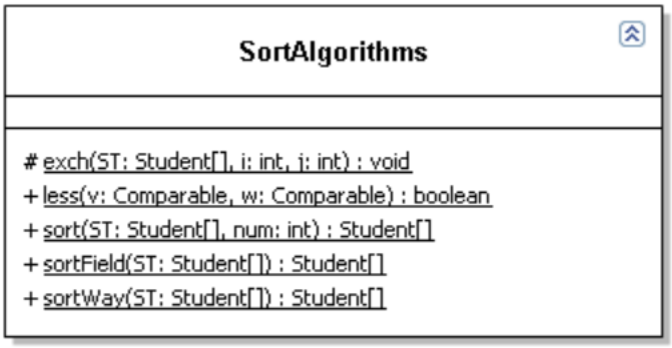
+course: String
+coName: String
+credit: String
+num: int
+GraCount: int[]
+highscore: int
+lowscore: int
+averscore: double

+Course()
+getCourseData(course: String) : void

SortAlgorithms Class

Description:

This file contains the SortAlgorithms class, which uses selection sort and Stack data structure to realize the function of sorting in ascending or descending order.



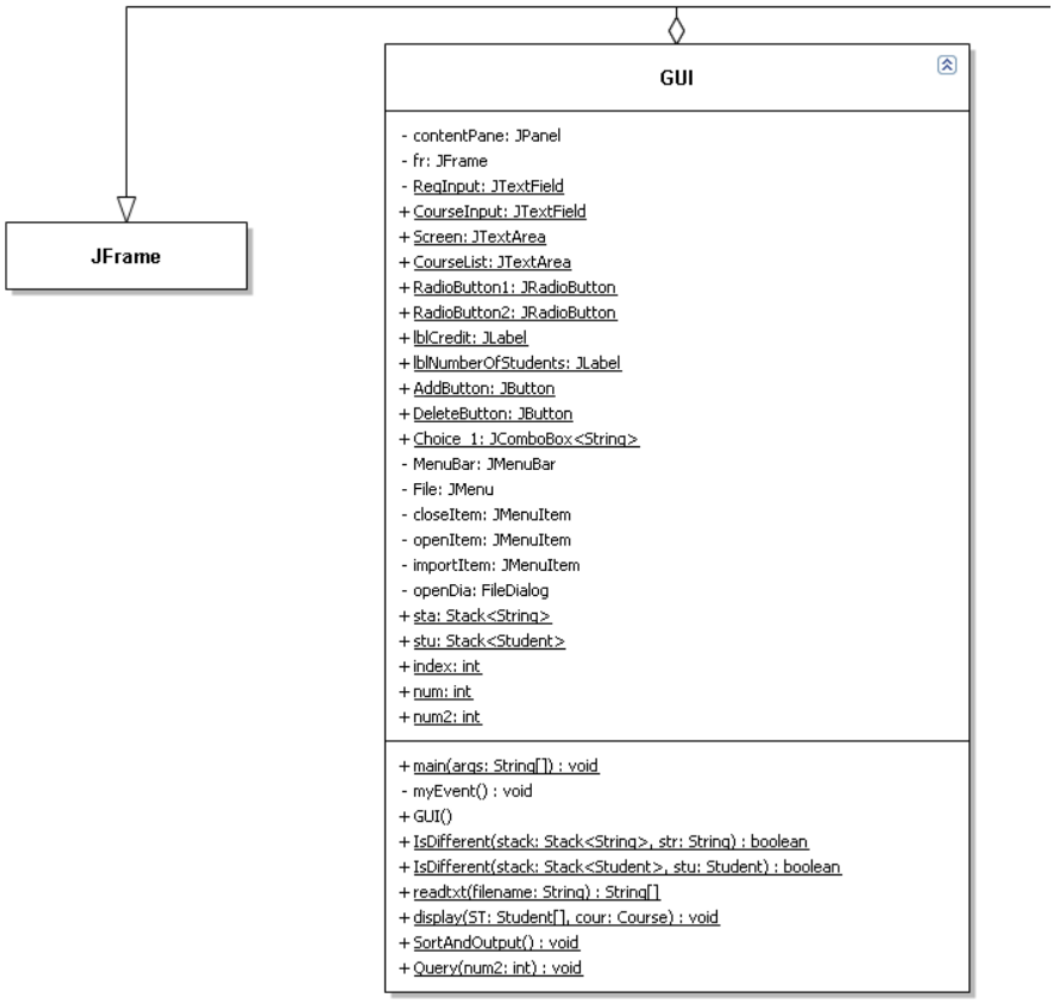
SortAlgorithms

#exch(ST: Student[], i: int, j: int) : void
+less(v: Comparable, w: Comparable) : boolean
+sort(ST: Student[], num: int) : Student[]
+sortField(ST: Student[]) : Student[]
+sortWay(ST: Student[]) : Student[]

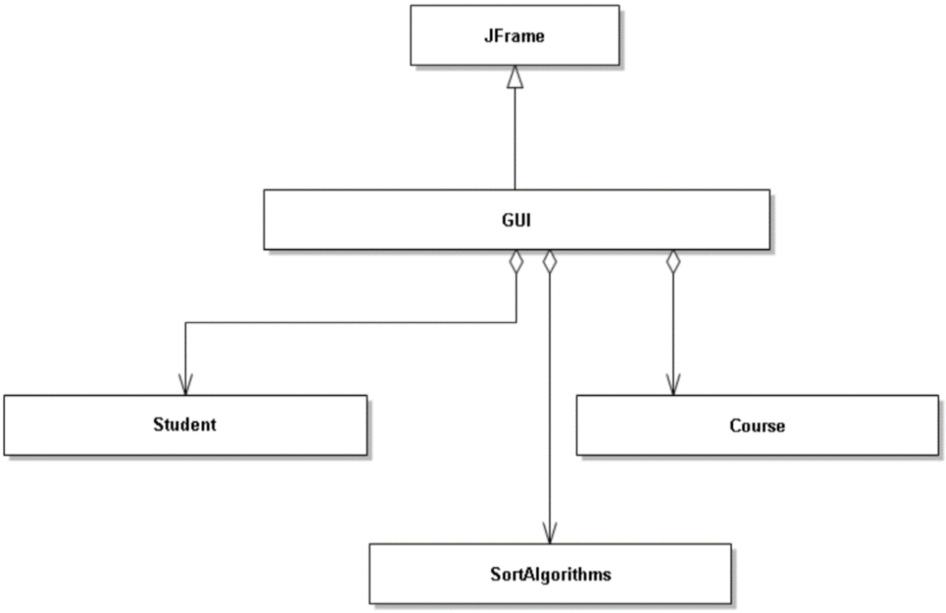
GUI Class

Description:

This file contains the GUI class, which contains varieties of GUI components that realize operation on graphic user's interface



General Relationship



3. Algorithm Design & Analysis:

Sorting: Selection Sort Complexity: O(N^2)

Searching: Sequential Search

Search by each inputted course individually Complexity: O(N^2)

Analysis: My program is based on **once-input store**, which means every time you input a course, the UGMS stores data of all the students who take this course at the same time. The number of one file data is less than 1000, so it is reasonable for a normal PC to execute for/while loop under 1 sec (Totally about 10^6, and 10^8 times / sec for normal PC).

4. Data Structures:

Basically, in Part II, I draw some diagrams to introduce the data structure of each class by using **NClass** software. Furthermore, I make a table to describe about DS, I denote that **"Data Structure"** as "DS".

Student Class

Member Name	DS	Description of storage
sur\giv\id\score\ Grade\GPA	Basic type (String\int\double)	The basic **one-course** data of each student containing surname, given name, id, score, Grade, GPA and so on
credit	Double array	The student's real credit of every course. (Parsed by Grade)
takecourse	String array	All the courses that the student takes
courseCredit	Int array	Corresponding credit of each taken course
Student()	/	Constructor

getsur()
getgiv()
getid()
getscore()
getGrade()
getStuData()
	/	Methods which are used to read data from the source file and return corresponding value

Grade2Credit()	/	Parse Grade to credit
Credit2Grade()	/	Parse credit to Grade
computeGPA()	/	Using credit/takecourse/courseCredit to compute the total GPA

Course Class

Member Name	DS	Description of storage
course/coName/credit/num	Basic type (String\int)	Basic data of a course

GraCount	Int array	Count the number of students of each Grade
highscore/lowscore/averscore	Int/double	Store the highest/lowest/average score of each course
Course()	/	Constructor
gerCourseData()	/	Read data from the source file and return corresponding value

SortAlgorithms Class

Member Name	**DS**	**Description of storage**
exchange()	/	Exchange two elements in an array
less()	/	Compare two elements (if former < latter return true)
sort()	/	Main sort algorithm
sortField()	/	Sort data in one aspect
sortWay()	Stack	Using stack to implement descent order

GUI Class

Member Name	**DS**	**Description of storage**
contentPane	JPanel	The JFrame window
fr	JFrame	The Frame
ReqInput	JTextField	The input of query
CourseInput	JTextField	The input of course name
Screen	JTextArea	The main screen

CourseList	JTextArea	The list of course
RadioButton 1&2	JRadioButton	The ascending & descending button
lblCredit	JLabel	The credit label
lblNumberOfStudents	JLabel	The label of the student number
AddButton&DeleteButton	JButton	Add&Delete course
Choice_1	JComboBox<String>	Sort choice
MenuBar	JMenuBar	Menu bar
File	JMenu	Menu name
closeItem/openItem/importItem	JMenuItem	Menu item
openDia	FileDialog	File dialog
sta	Stack	Store the course
stu	Stack	Store the Student element
index	int	Used to store course name
num	int	Index of choice 1
num2	int	Index of choice 2
main()	/	Start of the whole program
myEvent()	void	Operation of the file
GUI()	/	Constructor (contains GUI components)
IsDifferent()	/	Judge whether the new element is repetitive or not
IsDifferent(override)	/	Judge whether the new element is repetitive or not
readtxt()	String array	Read the raw data and split it into groups
display()	/	Reveal the data
SortAndOutput()	/	Execute sort and display method

Query()	/	Based on ReqInput, search for detailed data

5. Summary (Special Features & Known Bugs):

SF:

1. Implement user-friendly GUI.
2. Nice output format (using JTable).
3. Implement some advanced query functions (such as Surname Wild Card, GPA ranging).
4. Allow the user to add and delete courses according to their requirement.
5. No need to import file to the root in advance.
6. The UGMS would remind you sometimes by showing your message window or check box.
7. Unnecessary to worry about sorting time (< 1 sec).

KB:

1. Nothing would happen if you enter incorrect info while querying.
2. The records of the inputted students would not be deleted even though you have deleted the course from the course list. But it has no influence to the result.

***PS:** The word using both **Bold** and Highlight notation is important.