King Abdulaziz University Faculty of Computing and Information Technology Computer Science Department

CPCS204, Fall 2019 Program 1
CS Project Management System
Assigned: Thursday, 19th Sep 2019 Due: Wednesday, 2nd Oct 2019

Purpose

1. Learn to implement a linked list for a real-world problem.
2. Review file I/O (input/output).

Read Carefully:

This program is worth 5% of your final grade.

WARNING: This is an individual project; you must solve it by yourself. Any form of cheating will result in receiving zero in the assignment.

The deadline for this project is Wednesday, 2nd Oct 2019 by 11:59 PM.

Note: Once the clock becomes 11:59PM, the submission will be closed! Therefore, in reality, you must submit by 11:58 and 59 seconds.

LATE SUBMISSION: You are allowed to make a late submission, but there is a penalty. If you submit within 24 hours of the due date (so on Tuesday by 11:59PM), you will receive a 25% deduction. If you submit within 48 hours of the due date (so on Wednesday by 11:59PM), you will receive a 50% deduction.

Blackboard Submission:

This project must be submitted online via Blackboard.

The source file(s) of your program should be zipped up. You must name the zip file using the following naming convention:
SectionNumber_StudentID_ProgramNumber.zip

Example: EA_1110348_P2.zip

Question:	1	2	3	Total
Points:	20	20	60	100
Score:				

Concept Application & Algorithmic Part

Background:

Computer Science Senior Projects: The senior project provides an integrated assessment of the students toward the desired software engineering competencies. The senior project is the first step to transfer the students from the academic community to the industrial environment. Students are expected to register for the senior project with a faculty member(s) whose specialty and interests are compatible with the preferred topic of the project. The senior project requires at least two semesters of work. Therefore, students will be allowed to register the course after completing CPCS223, CPIS334, CPCS351, CPCS241, CPCS361, and CPCS331 courses.

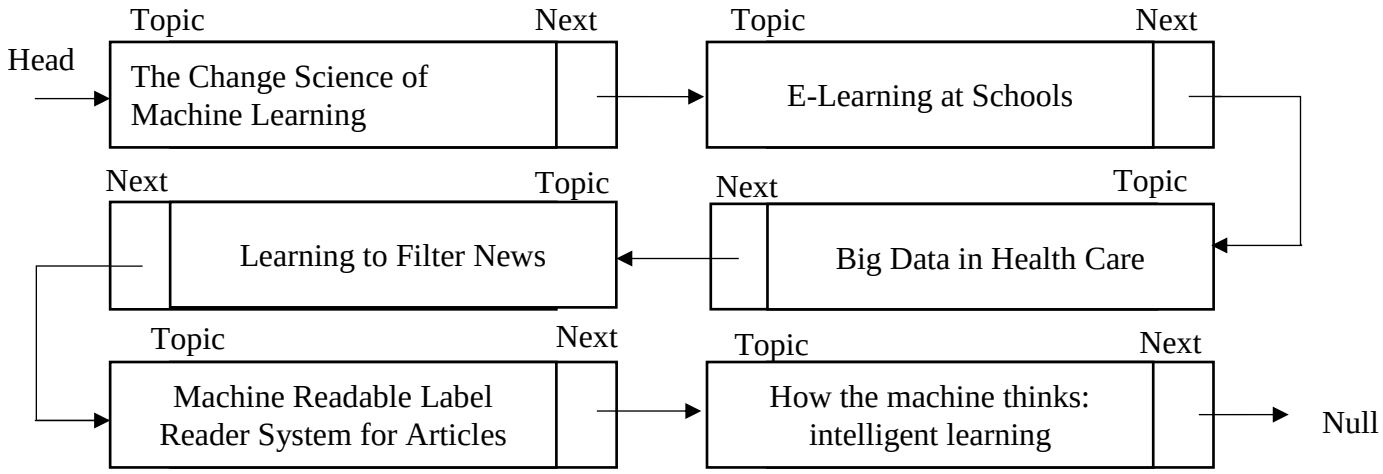
Implementation of the CS Senior Project management system in the CS department is done using the following procedures:

1. The senior project supervisors will provide their research interests from different specializations.
2. Each student in the CS program will nominate at least one faculty member to work as students' supervisor.
3. The nominated senior project supervisors will meet as early as possible to discuss the proposed topic of the project.
4. Each group shall provide a clear course description to check the minimum requirements to register in the senior project according to their study plan.
5. Final approval from the supervisors is required to enroll students in the chosen specialization.

Question 1: (20 points)

1. **Concept Application:** Suppose a linked list contains following topics of the CS projects. Write the steps to delete the **"E-learning at School"** from this list by showing
   - the **position of the pointer** and

the status of the **linked list** after every step.

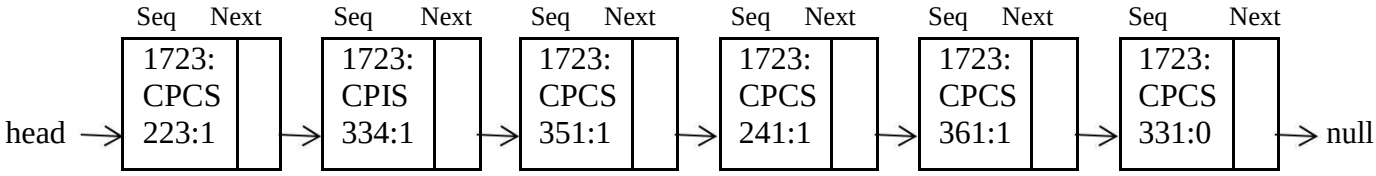


What are the steps used to delete the Node **"E-learning at School"**:

Question 2:(20 points)

2. Algorithm Write up: Suppose a linked list contains Student ID, CS courses, and current status. **Write an algorithm** which must check and print whether the student complete the minimum requirements to register in a senior project. While the student ID and the CS course contains string values, and current status contains a flag '1' if the student completes the course or '0' if the student did not complete the course.

For example, if the linked list contains the following nodes



The output of the algorithm should be

```
The student ID:1723 has not completed the requirements.
```

Algorithm:

Input:

Output:

Method:

Question 3:(60 points)

Program 2: CS Project Management System

Objective

The primary objective of this program is to implement a linked list. The secondary objective is to practice with File I/O.

Program Description

Write a program to manage the registration in CS senior project. We have a file for a senior project, each project contains (student ID, Student Name, research interests, suggested topic, an array of the previous courses including status, approval, supervisor ID). A student can enroll in the senior project if he/she find a suitable supervisor, complete a minimum requirement of the studied courses.
We have another file with supervisor information, each contains (Supervisor ID, Supervisor Name, an array of the research interests).

1. Write a function to add a new student (making sure they don't exist already) to a student file.
2. Write a function to check the course requirements status
3. Delete a student by ID
4. Delete all student from the linked list above that have not completed the minimum requirement.
5. Create a linked list for the registered student with **ascending** order.

Student ID	**name**	**Research interest**	**Suggested topic**	**courses**					**approval**	**Supervisor ID**	**next**
1723	Asma	Artificial intelligent	How the machine thinks: intelligent learning	CPCS223	CPIS334	CPCS331		….	true	00023	
				1	1	1					

node

Your program will take new student information and new supervisor information to store it in the file. The program then read a text file as input (student.txt) which the information will be retrieved. The basic functionality of CS senior project management system will be implemented using a **linked list**. The basic algorithm is as follows:

Read the student text document
For each student in the file
    If the courses status is not confirming that the student completed the minimum requirements
    Remove the student from the list
    Else
    Read the supervisor text document

Check the student research interest whether it matches any of the available research interests and return approval
Print the final list of all students achieve both conditions: complete the minimum requirements and has a supervisor

- To read the document use the Scanner() method
- To iterate over each student or supervisor in the documents use Scanner.next() method
- To add the student to the list, create a new node with the student information, and check his/her status.
- To check the status of the courses , use the *check_Course* method**.**

```
Public boolean check_Course ( String courses) {
..
...
}
```

- To check the supervisor availability, use the *SupervisorStudentInterests* method

```
Public boolean SupervisorStudentInterests ( String researchInterest) {

        Boolean Available=false;

                return Available;
}
```

- The system only allows student with active status to add their research topic.
- To remove a student by using the student ID, check the supervisor ID, and delete each supervisor of this student that doesn't have another student.

The user should select from the menu. **The menu includes add student, add supervisor, add research topic, print student information, print supervisor information, and print senior project list.**

Implementation

For this program, you will create the following classes:

- *student*.java: This class will be used to create objects of type student. Each student object will store the student ID, Student Name, research interests, suggested topic, an array of the previous courses including status, approval, supervisor ID frequency in the document.

- supervisor.java: This class will be used to create objects of type supervisor. Each supervisor object will store the Supervisor ID, Supervisor name, an array of the research interests in the document.

- *SeniorProjectSystem.java*: It will be used to create a linked list with nodes of type *student*. All the methods will be implemented in this class.

- *StdMenue.java*: This is the class that will contain the main. This class also print the menu, and the user should select his option.

Input File Specifications

You will need to add a new student to the file and read a current information from text file. This should be automated when the user selects to add a new student or select to display the current list.

Student.txt: This is the text file from which the status will be checked.

Let's assume that the student.txt consists of the following information:

1723, Asma, artificial intelligent, How the machine thinks: intelligent learning,1,1,1,1,1,1,true,00023#
1777,Rania, database, nan,1,1,1,0,0,0,false,0#
1003,Sara, network, nan,1,1,1,1,1,1,true,00013#
1743,Roaa, artificial intelligent, nan,1,1,1,0,0,0,false,0#

Supervisor.txt: This is the text file from which the status will be checked.

Let's assume that the supervisor.txt consists of the following information:

00023, Dr. Haneen, artificial intelligent, data mining, pattern recognition#
00013, Dr. Manar, database, network#
00011, Dr. Hajar, software engineering, games#

The menu to be implemented are as follows

The user will be allowed to reselect from the menu until press **7** to exit.
The menu includes add a new student, print supervisor list, add research topic, print student information, print supervisor information, remove the student and print senior project list.

1. **Add a new student.**
2. **Print supervisor list.**
3. **Print student list.**
4. **Add research topic.**
5. **Remove student.**
6. **Print senior project list in ascending order**
7. **Exit.**

Example Input:

1. Add a new student.

This selection will allow the user to enter the student information using student.java and store the information in student.txt. Note: the system should check whether the student **already exists** in the list.

Example output:

```
Please enter the student information:

student ID:1333
Student Name: Areej
research interest: software engineering
 courses: CPCS223, CPIS334, CPCS351, CPCS241, CPCS361, CPCS331
The student is added!
```

```
Please enter the student information:

student ID:1723

The student already exists!
```

Example Input:

2. print supervisor list

You will need to read a text file. This should be automated. Do not ask the user to input these files.

Example Output:

```
Available Supervisors:
```

Supervisor ID	Name	Research interests		
00023	Dr. Haneen	artificial intelligent	data mining	pattern recognition
00013	Dr. Manar	database	network	
00011	Dr Hajar	software engineering	games	

Example Input:

3. Print student list.

To display the student information, read the stored information from the student.txt file

Note: you have to display the course status in a table format, you might need `printf.`

Example output:

Student ID	Research interest	Suggested topic	courses						Approval	SupervisorID
1723	Artificial intelligent	How the machine thinks: intelligent learning	CPCS223	CPIS334	CPCS351	CPCS241	CPCS361	CPCS331		
			1	1	1	1	1	1	true	00023
1777	database		1	1	1	0	0	0	false	0
1003	network		1	1	1	1	1	1	true	00013
1743	Artificial		1	1	1	0	0	0	false	0

	intelligent									
1333	software engineering		1	1	1	1	1	1		

Example Input:

4. Add research topic

This option will read the student information from the list based on the matched student ID. The system will call the function to check the courses status *check_Course and* update the *Approval* information on *student.txt*. If the *approval* status changed to *true* the system will ask to add the suggested research topic and update the *student.txt.*

Example Output:

```
Please enter the student ID: 1333

The student studied all required subject, and please enter the

topic: Machine readable label reader system for articles
```

Student ID	Research interest	Suggested topic	courses						Approval	SupervisorID
1333	software engineering	Machine readable label reader system for articles	CPCS 223	CPIS334	CPCS 351	CPCS 241	CPCS 361	CPCS 331		
			1	1	1	1	1	1	**true**	00011

Example Input:

5. Remove student

This option should display the menu:

1- Remove student by his/her ID

2- Remove students that have not to complete the minimum requirement.

This method will print the student information from the student.txt file. Then request to enter the choice. If the choice is 1, the system will ask about the student ID that you would like to remove. If the choice is 2, the system should display only the removed student.

Example Output:

Student ID	**Research interest**	**Suggested topic**	**courses**						**Approval**	**SupervisorID**
1723	Artificial intelligent	How the machine thinks: intelligent learning	CPCS 223	CPIS 334	CPCS 351	CPCS 241	CPCS 361	CPCS 331		
			1	1	1	1	1	1	true	00023
1777	database		1	1	1	0	0	0	false	0
1003	network		1	1	1	1	1	1	true	00013
1743	Artificial intelligent		1	1	1	0	0	0	false	0
1333	software engineering		1	1	1	1	1	1	true	

```
Please select your option:

   1. Remove student by his/her ID
Remove students that have not to complete the minimum
requirement.
Please enter the student ID that you would like to remove: 1777
```

Student ID	Research interest	Suggested topic	courses						Approval	SupervisorID
1777	database		CPCS223	CPIS334	CPCS351	CPCS241	CPCS361	CPCS331		
			1	1	1	0	0	0	false	0

2. Remove students that have not to complete the minimum requirement.

Student ID	Research interest	Suggested topic	courses						Approval	SupervisorID
1777	database		CPCS223	CPIS334	CPCS351	CPCS241	CPCS361	CPCS331		
			1	1	1	0	0	0	false	0
1743	Artificial		1	1	1	0	0	0	false	0

	intellig ent									

Example Input:

6. Print senior project list in ascending order

The method will search for an only approved student and has a supervisor.

Example Output:

Student ID	Student Name	Research interest	Suggested topic	Supervisor ID
1723	Asma	Artificial intelligent	How the machine thinks: intelligent learning	00023

Sample Input & Output File

You must print your output to an output file called **`output.txt`**. We have provided you a sample input with matching output.

*****WARNING*****

Your program MUST adhere to the EXACT format shown in the sample output file (spacing capitalization, use of dollar signs, periods, punctuation, etc). The graders will use large input files, resulting in large output files. As such, the graders will use text comparison programs to compare your output to the correct output. If, for example, you have two spaces between in the output when there should be only one space, this will show up as an error even though you may have the program correct. You will get points off if this is the case, which is why this is being explained in detail. The minimum deduction will be 10% of the grade, as the graders will be forced to go to text editing of your program in order to give you an accurate grade. Again, your output MUST ADHERE EXACTLY to the sample output.

Grading Details

Your program will be graded upon the following criteria:

1) Adhering to the implementation specifications listed on this write-up.
2) Your algorithmic design.
3) Correctness.
4) **Use of the three classes, as specified. If your program is missing these elements, you will get a zero. Period.**
5) The frequency and utility of the comments in the code, as well as the use of white space for easy readability. (If your code is poorly commented and spaced and works perfectly, you could earn as low as 80-85% on it.)
6) Compatibility to the **newest version** of NetBeans. (If your program does not compile in NetBeans, you will get a large deduction from your grade.)
7) Your program should include a header comment with the following information: your name, **email**, account number, section number, assignment title, and date.
8) Your output MUST adhere to the EXACT output format shown in the sample output file.

Deliverables

You should submit a zip file with four files inside:

1. *ConceptPart.doc (Containing concept and algorithm parts)*
2. *student.java*
3. *supervisor.java*
4. *SeniorProjectSystem.java*
5. *StdMenue.java*

***These two files should all be INSIDE the same package called **`CSProjectManagmentSystem`**. If they are not in this specific package, you will lose points.

NOTE: your name, ID, section number AND EMAIL should be included as comments in all files!

UML Diagrams:

For this program, you will create **four** Classes (UML diagram shown below):

student

Data Members
`private String studentID;`
`private String studentName;`
`private string research_intrest;`
`private string Topic;`
`private int [6] course;`
`private boolean approval;`
`private String supervisorID;`
`private student next;`

Operations/Methods
`student() //` one or more Constructors
`check| std()`
 	
And any other methods you need.

supervisor

Data Members
`private String supervisorID;`
`private String supervisorName;`
`private String [3] Intrest;`

Operations/Methods `supervisor()//` one or more Constructors And any other methods you need.

SeniorProjectSystem.java

Data Members
`private student head;`
`private superviosr head;`

Operations/Methods
`check_Course ()`
`SupervisorStudentInterests()`
`RemoveStudent()`
`PrintStudent()`
`PrintSupervisor( )`
`PrintRegisteredStuentList()`
 	
ALL necessary methods for linked-list operations

StdMenue
Data Members `As needed`
Operations/Methods `public static void main()` ALL necessary methods for linked-list operations