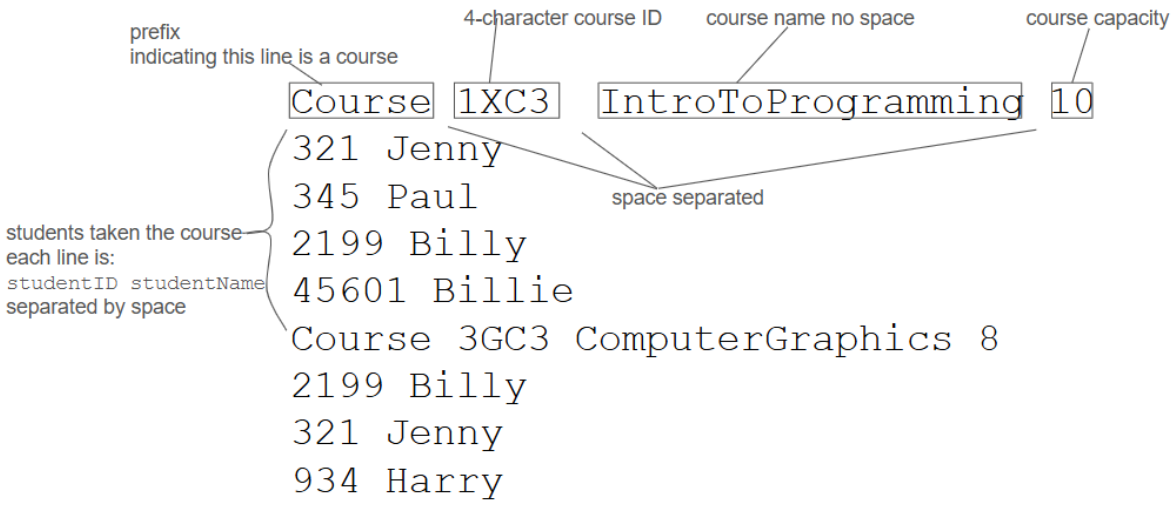
COMPSCI 1XC3 C01,C03
Assignment 3
(100 points)
Due date: March 24th, 2025

In assignment 3, you are provided with *main.c*, *course.h*, *student.h*, and an example text file *courseInfo.txt*.

Format of Input Text File

The example of the input file is provided in *courseInfo.txt*. The format is as follows:



Functionality of main.c

In *main.c*, it contains the general functionality of the assignment. You will read in the formatted text file, store the information in data structures defined in *course.h* and *student.h,* and write out student information to a formatted output file "studentInfo.txt".

```
int main(int argc, char** argv)
{
  readCourseInfoFrom("courseInfo.txt");
  writeStudentInfoTo("studentInfo.txt");
  cleanup();
  return 0;
}
```

Course data in course.h

In *course.h*, four parallel arrays are defined:

```
char* courseIDs [MAX_COURSE_NUM];
char* courseNames [MAX_COURSE_NUM];
int courseCapacities [MAX_COURSE_NUM];
int* courseTakenByStudents [MAX_COURSE_NUM];
```

courseIDs[i], courseName[i], courseCapacities[i] and courseTakenByStudents[i] describe the id, name, capacity and ids of students for course i.

Student data in student.h

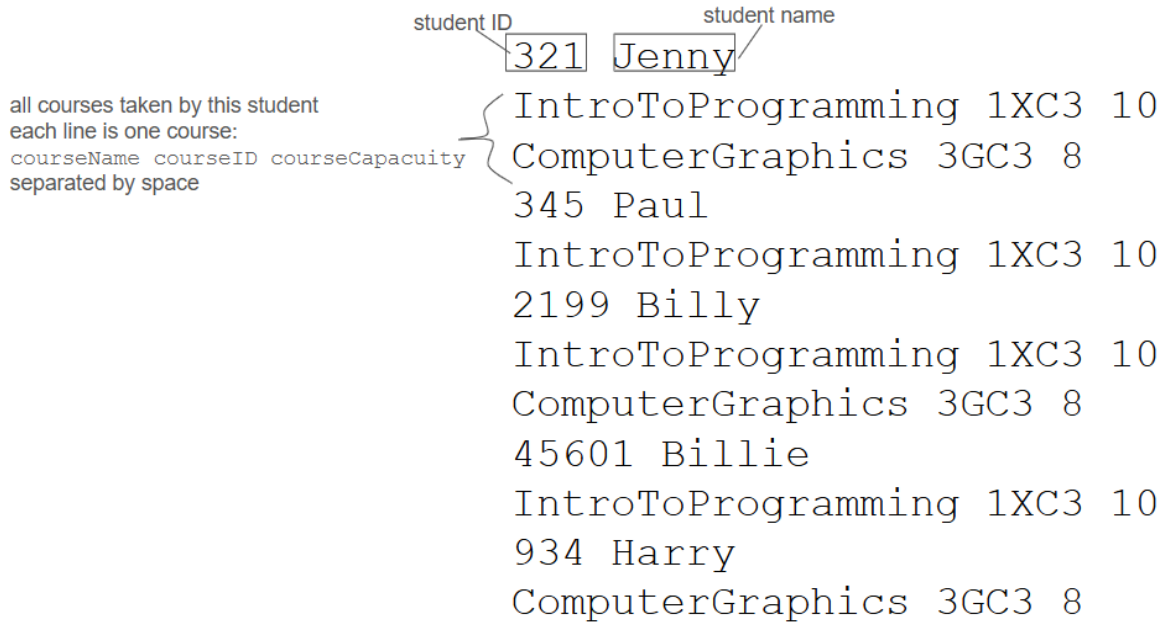
In *student.h*, three parallel arrays are defined:

```
int studentIDs [MAX_STUDENT_NUM];
char* studentNames [MAX_STUDENT_NUM];
char** studentTakesCourses [MAX_STUDENT_NUM];
```

studentIDs[i], studentNames[i], and studentTakesCoruses[i] describe student i's id, name, and courses ids taken by student i.

Format of Output Text File

Once the information is read, *main.c* will write out the student information to *studentInfo.txt*, examples provided. The format of the output file is as below. In the output file, the order of students and the order of courses follow the appearance orders of students and courses in the input file. From the *courseInfo.txt* example, student order in the output *studentInfo.txt* should be: Jenny, Paul, Billy, Billie and Harry; for each student the course order is 1XC3 then 3GC3 if they take multiple courses.



Note:

- Put your implementation of file reading in function `void readCourseInfoFrom(char* fileName)` 20' and your file writing in function `void writeStudentInfoTo(char* fileName)` 20'.
- Please make sure that `malloc`ed space is properly released using `free` without memory leak in `void cleanup()` 20'.
- Note that coding style e.g. tidiness and readability may take up minor points, up to 5' among the above 60'.
- You can assume the input file format is correct, and each student maximally takes 5 courses. Students may have the same name, but their student ids are unique. Course ids are also unique.
- Please do not change any of the header files, and make sure your *main.c* compiles correctly with *course.h and student.h* using gcc main.c .
- Please submit your *main.c* file and TA will recompile your submitted code and test run during grading 40'. Pay attention to the order of students and courses in the output file, as wrong orders could fail the test cases and lead to points lost. Please verify with the example *studentInfo.txt*

You may feel the trouble managing the parallel arrays. We will update to using structure in our next assignment.