PROG20799: Data Structures and Algorithms in C

Evaluation: **10 points, 7.5%** of your final grade.
Due date: See SLATE.
Late submission: 10% per day penalty, up to 3 days.

In this assignment you need to create a linked list of structures to store student's info. Your goal is to collect and sort students' records by GPA, and print them in a table-like output.
You must use the program template **main.txt** file and function prototypes given to you.
Please check the demo version: https://youtu.be/i-DKOCZVX7s
You must use a **single linked list** with the following **structure**:

```
typedef struct node {
   size_t id;
   char* name;
   float gpa;
   struct node *next;
} student_t;
```

Requirements:

- Please download and use **main.txt** template file.
  You must only use this template file with function prototypes or you'll get a **zero grade**.
- You absolutely must use the structure mentioned above.
  You'll get a **zero grade** if you use anything else to store student's info.
- Structures must be allocated on the **HEAP**. You'll get a **zero grade** otherwise.
- You must set initial **student id** to **1000** and auto-increment it when a new student is added.
- You must use **fgets()** to get both the **name** and **gpa** of the student (check MAX_LEN macros).
- You must allocate student's name on the HEAP and use appropriate **strto…()** function to convert input to floating-point value. Do NOT use **scanf(), sscanf()** or **atof()** anywhere!
- You must use **Insertion into Sorted Linked List** algorithm when you insert a new node (with student's info) into the **linked list**: http://babanski.com/files/prog20799/assignment3/
- You cannot modify the **main()** function given to you in the template file.
- You must use and implement the following functions (see template file):

```
student_t* createNode();          // function to create the node
student_t* createList();          // function to create the list
void displayList(……..);           // function to display the list
void insertNode(.....);           // function to insert node into the sorted list.
void removeList(.....);           // function to delete the linked list
```

- You must destroy the linked list and deallocate memory at the end of the program.
  Please use function **removeList(..)** accordingly.
- Your program must work similarly to the Demo version posted on Youtube, ignore the incorrect input, have same error notifications, sort the records, etc.
- Your solution should have optimal time and space complexity.

Hints:

1. The reason we want to use a linked list instead of an array is because you can “sort” the linked list during console input. I.e. you create a new node, get student’s GPA and insert the node into the linked list in such a way that it is always sorted by students’ GPA.
   Array of structures can't be used because you don't know how many students you have and then you can't sort it on-the-fly as easy.

2. When you enter empty name then it is assumed there are no more nodes to add to the list.

3. Function **createNode()** must collect student’s info from the console, allocate memory on the HEAP for the node and for the name, and return memory address of the node.

4. In the **removeList()** function you need to de-allocate memory for the name first and then for the node itself.

5. You should allocate memory for student’s name on the HEAP using **calloc()** function.
   Don’t forget to use **strlen(name)+1** as size.

6. Do not forget about special cases in **Step 5**: http://babanski.com/files/prog20799/assignment3/

7. Check the demo posted on YouTube. Demo input and output:

```
Creating List of students:

Insert student's name (id=1000): Darth Vader
Insert student's GPA (0-100): 55.7

Insert student's name (id=1001): Princess Leia
Insert student's GPA (0-100): 77.5

Insert student's name (id=1002): Emperor Palpatine
Insert student's GPA (0-100): 92

Insert student's name (id=1003): Chubaka
Insert student's GPA (0-100): 5

Insert student's name (id=1004): Kylo Ren
Insert student's GPA (0-100):
Incorrect value of GPA! Ignore student input!

Insert student's name (id=1004): Kylo Ren
Insert student's GPA (0-100): -4
Incorrect value of GPA! Ignore student input!

Insert student's name (id=1004):

============== List of Students: ==============
  No      GPA     ID        Name
-----------------------------------------------
  1.     92.00   1002    Emperor Palpatine
  2.     77.50   1001    Princess Leia
  3.     55.70   1000    Darth Vader
  4.      5.00   1003    Chubaka
```

Submission:

This is an individual assignment. Even partially copied code will be subject to regulations against academic integrity. Do NOT discuss or share your solution with anybody. Posting this assignment or solution on the Internet is a violation of the Student Code of Conduct.

- Please make sure your code is POSIX-compliant (works in NetBeans/Cygwin)!
- Your submission must be one **text file only**: **assignment3.txt**
- Please copy **main.c** as text file **assignment3.txt** with extension **.txt**
- You must upload **assignment3.txt** file to the Assignment Dropbox by due date.
- Verify that **assignment3.txt** file you submitted can be opened by Notepad in Windows.
- You'll get **-3 points** if you submit incorrect file (.zip file, or file with incorrect name/extension).
- Your submission must be unique or have references.
- Please **self-evaluate your code** in the comments section of the Dropbox.
- **Late submissions are penalized 10% / day (1 point / day), up to -3 points.**
  Submissions won't be accepted after 3 days.

Grading Scheme:

- See **Main requirements** and also **Course_Introduction.pdf**.
  Deductions will be applied if partial functionality is provided.
- You'll get **zero grade** if your code doesn't compile.
  Compilation warnings are considered to be major mistakes.
- Comments are not required but recommended. However, "debugging code" or commented out "old code" is a minor mistake (unless it's clearly stated in the comments that you want me to take a look at such code).
- Please check submission requirements for any additional deductions.