Raster Graphic Project in C

Due Time: 23.59, Sun 23 September 2018 **Earnings:** 4% of your final grade

NOTE: Plan to finish a few days early to avoid last minute hardware/software holdups for which no allowance is given.
NOTE: The code in this assignment must be your own work. It must not be code taken from another student or written for you by someone else, even if you give a reference to the person you got it from (attribution); if it is not entirely your own work it will be treated as plagiarism and given a fail mark, or less.

Purpose: You are to write the code in Visual Studio 2015 for a simple C language console application that holds the data of a raster graphic application (there is no actual graphics in the assignment) using a forward list (aka singly-linked list) in dynamic memory for its data. This will give you an opportunity to review material that has already been taught in an earlier C course and get up to speed for programming that is used in the course. In the lab you will be shown how to set up a Visual Studio 2015 project to build and run the application.
Part of the code is shown on the next page; it is also on Brightspace in a text file that you can copy and paste. You **MUST** use this code **without modification (not a single character changed): no code added or removed, no new global variables, no macros, no defines and no statics**. Your task is to implement in C after the main() function only the functions that are declared at the top of the ass0.c file and not add any new ones. All your code is in the file ass0.c.

The RasterGraphic is a series of GraphicElements held in a forward list. When the list runs it displays the details of each GraphicElement at intervals of 1 second using the system clock (you are given the code for this).
You can:

- add a new GraphicElement to the RasterGraphic at a position selected by the user,
- delete the first GraphicElement in the list,
- report the RasterGraphic to show the list of GraphicElement details one after another at 1 second intervals (code supplied),
- quit leaving no allocated dynamic memory (i.e. no memory leak),

An example of the output of the running application is given at the end. Yours must work identically and produce identical output.

Note the following:

- the file you submit must be named ass0.c,
- dynamic memory management is done with malloc() and free(),
- you can only use functions like strlen() and strcpy() or similar etc. from the standard C library to handle strings of null terminated char arrays,
- when the application terminates it releases all dynamically allocated memory so it does not have a memory leak (or you lose 30%).

See the Marking Sheet for how you can lose marks, but you will lose at least 60% if:
1. you change the supplied code in any way at all (not a single character) - no code added (except header includes) or removed, no macros, no defines, no statics and no additional global functions or variables,
2. it fails to build in Visual Studio 2015,
3. It crashes in normal operation (such as running an empty list etc.),
4. it doesn't produce the example output.

Part of the code is shown on the next page. You MUST use this code without modification**.** Your task is to add the implementation of the declared functions using the style of the posted Submission Standard. All the code is in a single file named ass0.c.

What to Submit: Use Brightspace to submit this assignment as a zip file (**not** RAR, not 9zip, not 7 zip) containing only the single source code file (ass0.c). The name of the zipped folder **must** contain your name as a prefix so that I can identify it, for example for CST8219 using my name the file would be tyleraAss0CST8219.zip. It is also vital that you include the Cover Information (as specified in the Submission Standard) as a file header in your source file so the file can be identified as yours. Use comment lines in the file to include the header.

Before you submit the code,

- check that it builds and executes in Visual Studio 2015 as you expect - if it doesn't build for me, for whatever reason, you get a deduction of at least 60%.
- make sure you have submitted the correct file – if I cannot build it because the file is wrong or missing from the zip, even if it's an honest mistake, you get 0.

There is a late penalty of 25% per day. Don't send me the file as an email attachment – it will get 0.

Example code (also in a text file on the LMS you can copy and paste). Don't change it.

```
// ass0.c
#define _CRT_SECURE_NO_WARNINGS
#define _CRTDBG_MAP_ALLOC     // need this to get the line identification
//_CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF|_CRTDBG_LEAK_CHECK_DF); // in main, after local declarations
//NB must be in debug build

#include <crtdbg.h>
#include <stdio.h>
#include <time.h>

typedef enum{FALSE = 0, TRUE} BOOL;

struct GraphicElement{
          char* fileName;
          struct GraphicElement* pNext;
};

struct RasterGraphic{
          struct GraphicElement* GraphicElements;
};

// Forward declarations
void InitRasterGraphic(struct RasterGraphic*);
void InsertGraphicElement(struct RasterGraphic*);
void DeleteGraphicElement(struct RasterGraphic*);
void ReportRasterGraphic(struct RasterGraphic*);
void CleanUp(struct RasterGraphic*);

int main(void)
{
          char response;
          BOOL RUNNING = TRUE;
          struct RasterGraphic RG;
          _CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF | _CRTDBG_LEAK_CHECK_DF);
          InitRasterGraphic(&RG);

          while (RUNNING)
          {
                    printf(   "MENU\n 1. Insert a GraphicElement\n 2. Delete the first GraphicElement\n 3. Report the RasterGraphic\n 4. Quit\n");
                    scanf("%c", &response);
                    switch (response)
                    {
                    case '1':InsertGraphicElement(&RG);break;
                    case '2':DeleteGraphicElement(&RG);break;
                    case '3':ReportRasterGraphic(&RG);break;
                    case '4':RUNNING = FALSE;CleanUp(&RG);break;
                    default:printf("Please enter a valid option\n");
                    }
                    printf("\n");
                    while ((response = getchar()) != '\n' && response != EOF);// clear input buffer
          }
          return 0;
}


void ReportRasterGraphic(struct RasterGraphic* pA)
{
          int counter = 0;
          time_t startsec, oldsec, newsec;
          struct GraphicElement* iterator = pA->GraphicElements;

          if (pA->GraphicElements == 0)
          {
                    printf("No GraphicElements in the RasterGraphic\n");
                    return;
          }
```

```
        printf("Report the RasterGraphic\n");
        startsec = oldsec = time(NULL);
        while (iterator)
        {
                newsec = time(NULL);
                if (newsec > oldsec)
                {
                        printf("GraphicElement #%d, time = %I64d sec\n", counter++, newsec-startsec );
                        printf("Image file name = %s\n",iterator->fileName);
                        iterator = iterator->pNext;
                        oldsec = newsec;
                }
        }
}
```

Example Output:

```
MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
1
Insert a GraphicElement in the RasterGraphic
Please enter the GraphicElement filename: GraphicElement_1
This is the first GraphicElement in the list

MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
1
Insert a GraphicElement in the RasterGraphic
Please enter the GraphicElement filename: GraphicElement_2
There are 1 GraphicElement(s) in the list. Please specify the position (<= 1) to insert at : 1

MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
1
Insert a GraphicElement in the RasterGraphic
Please enter the GraphicElement filename: GraphicElement_3
There are 2 GraphicElement(s) in the list. Please specify the position (<= 2) to insert at : 0

MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
3
Report the RasterGraphic
GraphicElement #0, time = 1 sec
Image file name = GraphicElement_3
GraphicElement #1, time = 2 sec
Image file name = GraphicElement_1
GraphicElement #2, time = 3 sec
Image file name = GraphicElement_2

MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
2

MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
3
Report the RasterGraphic
GraphicElement #0, time = 1 sec
Image file name = GraphicElement_1
GraphicElement #1, time = 2 sec
Image file name = GraphicElement_2

MENU
 1. Insert a GraphicElement
 2. Delete the first GraphicElement
 3. Report the RasterGraphic
 4. Quit
```