COMP 451

FORMAN CHRISTIAN COLLEGE

(A CHARTERED UNIVERSITY)

COMPILER CONSTRUCTION

Programming Assignment 1

It's an open books and open notes assignment. Use of Internet is allowed. This assignment should be done *individually*. You CANNOT share your code with any other student. We will apply turn it in on your code and check for a similarity index greater than 60%. In such a case both the scripts will be awarded ZERO marks.

Note that you can get help from the Internet, but should not copy paste the code.

Grading Criteria

Working Code: 60%

Properly formatted Report: 20%

Viva: 20%

Important: You need to submit a well formatted and well written report for this assignment. The report should carry following sections:

- Introduction about the problem in hand especially well written information about the preprocessor and its functions.
- Detailed and easy to understand description of your logic. Make separate section for each component. In this assignment at least three sections describing each function.
- Start early. **NO additional time in any case what so ever will be granted. A penalty of 60% will be applied on late submission.**
- **Submissions late by 24 hours will not be considered for grading.**
- Viva will be conducted for this assignment. Date will be announced later.

Hard Deadline: Code file should be submitted on or before 11:59 pm 24 Mar, 2024, on MOODLE course page. A hard copy of the report should be submitted on Mar 25, 2024 in class.

Your submission on MOODLE should carry

- **The code file.**
- **All the output files. (intermediate output files as well as the final output file)**
- **Your report. Report should be like documentation. Clearly describing your work. Marks for report are based on this.**
- **Make sure that the code file must carry the name and roll number as data dictionary (before start of code).**
- **ZIP all the files that you submit and name it with your roll number. Submit this zip file on Moodle. DONOT submit using email.**
- **You will be called for a viva based on what you have submitted.**
- **Note that you MUST NOT use any string manipulation functions. In other words, your code should not carry #include <string.h> statement. Any such code with string library calls can have a maximum 20% marks in total.**

COMP 451

Assignment Task [60 Marks]

In this task we will write a preprocessor. Your program should accept a C file from command line. The file should contain a valid C program. You need to write a single C file for this assignment.

Your program should accept a valid c program file as input provided by user on the command line.

Your program should perform the following tasks:

- Your program should first display the input file on console.
- Write a function **`void removeBlankLines(. . .)`** that should read the input file and removes any blank lines in the code. This function should write the output (the C file without blank lines in a file).
- Next write a function **`void removeComments(. . .)`** that accepts the output file from **`removeBlankLines()`** function and should remove any double slash or star slash comments from the file. It should write the file in another file.
- The file obtained from **`removeComments()`** function is now provided to **`void macroExpansion(. . .)`** function as input. This function will look for any macros (one or more) in the input file and should expand these. Macro expansion means that
    - The macro definition lines are removed.
    - Wherever in the code the macro head is used, should be replaced by macro body.

  The output file of this program should be written in a file named out.c, and should also be displayed on console.

The general structure of your code should look like this:

```
#include <stdio.h>
#include <stdlib.h>
//other include files or global variables or function prototypes go here
int main(int argc, char *argv[])
{
      //your logic for checking input arguments
      //other housekeeping stuff may appear here
      removeBlankLines(. . .);
      removeComments(. . .);
      macroExpansion(. . .);



      return 0;
}
void removeBlankLines(. . .)
{
//your logic for removing blank lines
}
void removeComments(. . .)
{
//your logic for removing comments
}

void macroExpansion(. . .)
{
//your logic for macro expansion
}
```

Few input file samples are shown for your convenience.

Input file 1

```
/*****in1.c****
*
*
*/
#include <stdio.h>
//defining macros


#define ON 1
#define OFF 0

void main()
{

      /*declaring variables*/
      int j = 2;
      int motor,sensorValue = 0;
      if(motor == ON)//what to do when motor is running
      {
            sensorValue++;
      }
      else if(motor == OFF)/* what to do when motor is not running */
      {
            sensorVlaue--;
      }


      return 0;
}
```

Final output for Input file 1

```
#include <stdio.h>
void main()
{
	int j = 2;
	int motor,sensorValue = 0;
	if(motor == 1)
	{
		sensorValue++;
	}
	else if(motor == 0)
	{
		sensorVlaue--;
	}
	return 0;
}
```

Input file 2

```
/*Program Name: in1.c
*Description:
      Input file for assignment 1
*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
//defining macros
#define MESSAGE1       "Hello class\n"
#define Message2       "Computer Science Department"

void main()
{

      //////////Print messages

      printf(Message1);
      printf(Message2);




      return 0;
}
```

Final output for Input file 2

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void main()
{
      printf("Hello class\n");
      printf("Computer Science Department");
      return 0;
}
```