Topic	Practical Assignment 4 Cover Sheet
Assignment Type	☒ Assessed ☐ Non-assessed ☒ Individual ☐ Group
Module	CSE101 Computer Systems
Due Date	December 6th, 2017 (Wednesday)
Student ID	1614649
Student Name	Kai-Yu Lu
Submission Date	2017.12.06

Declaration on Plagiarism and Collusion

I have read and understood the definitions of plagiarism and collusions as described in the University's Code of Practice on Assessment. As such, I certify the work presented in this report/assignment has been written solely by me and in my own words (except where references and acknowledgments are clearly defined). I agree to accept disciplinary actions should I be caught with the serious offence of plagiarism and/or collusion.

Signature: _____Kai-Yu Lu 卢凯郁_____

	Date Received	**No. of Days Late**	**Penalty**
For Academic Use			

Program Listing

```
#include "stdafx.h"
#include"string.h"
#define max 100

int main(int argc, _TCHAR* argv[])
{
  char inputName_Message[] = "Please enter your name: ";
  char reversedName_Message[] = "Your name in reverse: ";
  char inputLength[] = "\nThe length of your name is %d";
  char invalidError[] = "\nInvalid operation";
  char bydError[] = "\nYour input length is beyond the maximum of the length";
  char end_Message[] = "\nEnd of program";

  int alength = 0;          //Beause "length" has the special function in programming, I use "alength" to avoid the mistake. It means the length of the input.

  char inputName[max];
  char Array[max];
  memset(inputName, 0, sizeof(inputName));
  memset(Array, 0, sizeof(Array));

  __asm {
  start:
        lea eax, inputName_Message;   //Load address of the string'inputName_Message' into eax.
        push eax;                //Address of string, stack parameter call.
        call printf;             //Use library code subroutine.
        add esp, 4;              //Clean 4 byte parameter off stack.
                                 //It can show the message "Please enter your name: "
```

```
        lea eax, inputName          //Load address of the string‘inputName' into eax.
        push eax                    //Address of string, stack parameter call.
        call gets_s                 //Here we use "get_s" because it can recognize the space or other signs. It is more functional than "scanf".
        add esp, 4                  //Clean 4 byte parameter off stack.

        mov esi, 0                  //Initialize esi to 0, because  I will use it later.
        mov alength, 0
        cmp inputName[esi], NULL;   //Compare the inoutName with NULL
        je inputError;              //If inputName = NULL,jump to "inputError"
                                    //Here reslize the function that if we input nothing.
        mov esi, 0                  //Initialize esi to 0, because  I will use it later.
    compareLength:
        cmp inputName[esi], NULL;   //Compare the inoutName with NULL.
        je count;                   //If inoutName = NULL, jump to "count".
        inc alength;                //Every time this loop runs, "alength" will add 1, therefore it can count how many times this loop run. How many times this loop runs is equal to the length of the inputName.
        mov eax, alength            //Here eax stores the times of the loop has run.
        cmp eax, 20                 //Compare the eax (running times of the loop) with 20(max). Here 20 is the number I set. Here realize the function that if the length of the input is beyond the set, it can jump to beginning.
        jg beyondError;             //If eax(alength) > 20,jump to "beyondError"
        inc esi;                    //Entire array move forward one unit.
        loop compareLength;
    inputError:
        lea eax, invalidError;      //Load address of the string‘invalidError' into eax.
        push eax;                   //Address of string, stack parameter call.
        call printf;                //Use library code subroutine.
        add esp, 4;                 //Clean 4 byte parameter off stack.
        call getchar;
        jmp start;                  //It can show the message "Invalid operation" and jump to the beginning.
    beyondError:
```

```
        lea eax, bydError;          //Load address of the string‘bydError' into eax.
        push eax;                   //Address of string, stack parameter call.
        call printf;                //Use library code subroutine.
        add esp, 4;                 //Clean 4 byte parameter off stack.
        call getchar;
        jmp start;                  //It can show the message "Your input length is beyond the maximum of the length" and jump to the beginning.
    count:
        mov ecx, alength;           //Here ecx stores the running times of "compareLength".

        mov esi, 0;                 //Initialize esi to 0, because  I will use it later.
                                    //From "compareLength" and "count", we could realize the funtion of counting the lenght of the inputName.
    pushIt:
        movzx eax, inputName[esi];  //Move each char to eax register
        push eax;                   //Push eax onto stack
        inc esi;                    //esi+1
        loop pushIt;                //Continue running "pushIt".

        mov ecx, alength;           //Reset lool counter with MAX_SZ
        mov esi, 0;                 //Reset index counter
    popIt:
        pop eax;                    //Pop char out from stack in reverse.
        mov Array[esi], al;         //Store char back into reversename.
        inc esi;                    //esi+1.
        loop popIt;                 //Continue running "popIt".

        lea eax, reversedName_Message;//Load address of the string‘reversedName_Message' into eax.
        push eax;                   //Address of string, stack parameter call.
        call printf;                //Use library code subroutine.
```

```
            add esp, 4;              //Clean 4 byte parameter off stack.


            lea eax, Array           //Move Array onto eax.
            push eax                 //Push eax onto stack.
            call printf              //Use library code subroutine.
            add esp, 4               //Clean 4 byte parameter off stack.
                                     //From "pushIt" and "popIt", we could realize the funtion of
reversing the inputName.
            mov eax, alength;        //Move alength onto eax.
            push eax;                //Address of string, stack parameter call.
            lea eax, inputLength;    //Load address of the string‘inputLength' into eax.
            push eax;                //Address of string, stack parameter call.
            call printf;             //It will take two parameters from the stack; printf(alength, & inputLength).
            add esp, 8;              //Clean 8 byte parameter off stack.
                                     //The length has been created and here can show the messge "The
length of your name is %d",
            lea eax, end_Message;    //Load address of the string‘end_Message' into eax.
            push eax;                //Address of string, stack parameter call.
            call printf;             //Address of string, stack parameter call.
            add esp, 4;              //Clean 4 byte parameter off stack.
                                     //It can show the message "End of program".
        }
        return 0;
    }
```