Systems Programming 1 – 300167

Tutorial and Lab Practice Three

This work **will** be marked and is due in your scheduled lab class the week commencing Monday 28th of March, and is worth **4%** of your assessment. You should hand in a hard copy of your answers and demonstrate your programs to you Tutor during your scheduled lab session.
Email submission is not accepted.

Tutorial

1. From your chosen C resource, read the chapters about C data structures, recursive function, function call, function pointers and command line argument.
2. Review the terminology introduced and concepts taught in Lecture 3.
3. List all of the possible operations on a stack. Write pseudo code for each of these operations.

Lab Practice

1. Write a recursive program to calculate and output the first n numbers of Fibonacci sequence. The sequence starts with 0, and therefore is defined as $F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2}$. That is, the numbers are in the following integer sequence (**1%**):

   $0, 1, 2, 3, 5, 8, 13, 21, 34, ...$
2. Modify the above recursive program in order to get the argument n by command line. (**1%**)
3. Write a program implementing basic stack operations. (**2%**)

   You should use the following definitions:

```
typedef char stackitem;

struct stack
{
    stackitem  d;
    struct stack *next;
};

typedef  struct stack ELEMENT;
typedef  ELEMENT   *POINTER;
```

   Your program should be able to perform the following operations , selected from a simple menu:

(a) Push one character onto the stack. Your function should have the prototype:
`void push(POINTER *Top,stackitem a);`
An example is given below.

```
void push(POINTER *Top, stackitem a)
/* put a into the top of the stack */
{
    POINTER temp;
    temp = malloc(sizeof(ELEMENT));
    temp->d = a;
    temp->next = *Top;
    *Top = temp;
}
```

(b) Push a string of characters into a stack. Your function should have the prototype:
`void push_string(POINTER *Top,char *string);`
Your implementation should make use of the `push()` function.

(c) Print the contents the stack. Your function should have the prototype:
`void print_stack(POINTER Top);`
It should not modify the stack in any way, only display its contents.

(d) Pop the top character from the stack. Your function should have the prototype:
`stackitem pop(POINTER *Top);`.
An example is given below.

```
void pop(POINTER *Top)
/* remove the top item */
{
    POINTER Top1 = *Top;
    if (Top != NULL)
    {
      *Top = Top1->next;
      free(Top1);
    }
   else
   printf("Empty stack.\n");
}
```

(e) Delete the stack, i.e. remove all items from the stack. Your function should have the prototype:
`void delete_stack(POINTER *Top);`
Your implementation should make use of the `pop()` function.

Optional Work **(For those students who wish to practise more.)**

Adding the following to your stack program:

1. a function to exchange the top two elements of the stack;

2. functionality to reverse the stack and print the reversed stack.