MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG213 Data Structures – Worksheet 6b

Stack ADT: Dynamic Implementation

Purpose:
This worksheet is concerned with dynamic implementations of stacks and you will be practicing the concept of abstract data types. You are not expected to complete the entire worksheet in class. Work on as many problems as you can; the remaining problems you can use for practice and to test and refine your understanding.

For the questions from 1 to 6, you have to use the given CNG213_Worksheet6b_Template.c file

1) Implement the abstract data type Stack using a linked list. You may use the following declaration for a dynamic stack:

```
struct Node{
   int val;
   struct Node *next;};
typedef struct Node StackRecord;
typedef StackRecord *Stack;
```

   The operations to be implemented include `CreateStack()`,`MakeEmptyStack()`,`IsEmptyStack()`, `PushStack()`, `PopStack()`, `topOfStack()`.

2) Write a corresponding main program to test your stack implementation by asking the user to continuously push and pop integer values to and from the stack, respectively.
3) Create a header file stack.h and a separate implementation file stack.c for it.
4) You will simulate an interactive program that allows users to draw polygons. A polygon consists of an a priori unknown number of points (x, y). The user may (1) add points to a polygon using the function `AddPoint()`, (2) remove points in the reverse order into which they were added using the function `RemovePoint()`, or (3) complete a polygon using the function `CompletePolygon()`. In order to be able to remove points from the polygon in reverse order, these points need to be stored in a stack. Points are pushed onto a stack each time the function `AddPoint()` is called and popped from the stack each time the function `RemovePoint()` is called.
5) Extend your program to allow for multiple number of polygons to be stored. The number of polygons is not known a priori. When the function `CompletePolygon()` is called, the data structure for the polygon must be inserted into a larger data structure.
6) Implement a function ShowPolygon() which displays the (x,y) points of all polygons.

7) Alphabetical order is a system whereby character strings are placed in order based on the position of the characters in the conventional ordering of an alphabet. Write a program which includes following;
   a) Create a header file stackStatic.h and a separate implementation file stackStatic.c for it. This will include static implementation of the stack.
   b) Create a header file stackDynamic.h and a separate implementation file stackDynamic.c for it. This will include dynamic implementation of the stack.
   c) a function IsOrderedPhrase () that takes a phrase as an input and prints if the words in the phrase is ordered alphabetically or not. This is a very simple function that uses a stack to check if the input phrase is ordered alphabetically or not.
      For example: if the phrase is `David went to village yesterday`, then words in this phrase is not ordered alphabetically since `went` comes before `to` according the alphabet.
   d) In main () function, you should test the IsOrderedPhrase () function both with static and dynamic implementations of stack (part a and b).

Please note that Stack implementations should be exactly same with the worksheet 6a and 6b.

Possible Sample Run:

```
Enter a phrase: David went to village yesterday
Words in this phrase is not ordered alphabetically
```

Possible Sample Run:

```
Enter a phrase: David go to village yesterday
Words in this phrase is ordered alphabetically
```