EED 1010 ALGORITHMS & PROGRAMMING

Name: Enes

Surname: Erten

Lab#10

Date: 27.04.2020

Lab Section: 4 in lab, 1 in theory

TASK 1: Write a program that asks the user to choose an option from the menu given below.

```
Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with address.
4 to quit.

=>
```

If choice 1 is entered, the program then takes an integer from the user and pushes it onto the stack. If choice 2 is entered, an integer value is popped off from the stack. If choice 3 is entered the stack is displayed in such a way that address of each node, the corresponding integer data and the address of next node are all displayed. A sample output screen is given below. Note that each node in the stack holds the address of the next node in its list member. If choice 4 is entered, program execution stops. If a number other than 1, 2, 3 or 4 is entered, "Invalid choice" is printed on the screen and the menu is displayed again.

The code:

```
#include<stdio.h>

#include<stdlib.h>

//include libraries


typedef struct stacknode{

        int num;

        struct stacknode *next;

}snode;

//defining new varaible type stacknode

//firs elemnts is holdin an integer data

//second one is pointing same varaible type

//change name by typedef to snode
```

```
typedef snode *stacknodeptr;
//change name by typedef stacknodeptr

void pushstack(stacknodeptr *a,int n);
int popstack(stacknodeptr *a);
int isempty(stacknodeptr a);
void printstack(stacknodeptr a);
//functions prototypes

//this functions take a number and add to do stack
void pushstack(stacknodeptr *a,int n)
{
        stacknodeptr a1;
   //dynamic memory allocation
        a1=(snode *)malloc(sizeof(snode));

        if(a1!=NULL)
        {
                a1->num=n;
                a1->next=*a;
                *a=a1;
        }
   //the push operation
   //copy the data a1 and assign a1 to a
}

//this functipns delete last out  element of stack
//and returns the number which is deleted
int popstack(stacknodeptr *a)
{
        stacknodeptr a1;
```

```
        int n;

        a1=*a;
        n=(*a)->num;
        *a=(*a)->next;
        free(a1);
        //the pop operation
        //keep *a on a1
        //go to *a next
        //delelte memory a1
        return n;
}

//this function checks th stack is empty or not
//if it is empty return 0 else returns 1
int isempty(stacknodeptr a)
{
        if(a!=NULL)
            return 1;

        else
            return 0;
}

//this functions prints stack
void printstack(stacknodeptr a)
{
        if(a==NULL)
            printf("The stack is empty!\n");

        else
```

```
	{
		printf("%-25s%-25s%-
25s\n","AdressOfTheNode","StackValue(data)","AdressOfNextNode");


		while(a!=NULL)
		{
			printf("%-25p%-25d%-25p\n",a,a->num,a->next);
			a=a->next;
		}
	}
}


main()
{
	stacknodeptr a=NULL;
	int c,data;
	//do while loop beccasue the program must
	//excute menu least one time
	do
	{
		printf("Enter your choice\n");
		printf("1 to place data on the stack.\n");
		printf("2 to place data off the stack.\n");
		printf("3 to display the stack with adress.\n");
		printf("4 to quit.\n\n=>");

		scanf("%d",&c);
		//take menu option from stdin
		switch(c)
		{//switch
			case 1:
```

```
        printf("Enter the data to be placed on : ");
        scanf("%d",&data);
        //take data from stdin
    pushstack(&a,data);
    //send data to the stack
        printf("The stack in tabular form is : \n");
        printstack(a);
        //print stack
        break;

case 2:
        if(isempty(a))
        {//check list is empty
                printf("%d popped up.\n",popstack(&a));

                if(isempty(a))
                {//check again becasue if stack has one elemnt this wontprint
                //because there is no stack left
                  printf("The stack in tabular form is : \n");
                  printstack(a);
            }
        }

        else
            printf("The stack is empty!\n");

    break;

case 3:

        if(isempty(a))
```

```
                    {//check stack is empty
                        printf("The stack in tabular form is : \n");
                    printstack(a);
                    }//print stack

                    else
                      printf("The stack is empty!\n");

                    break;

            case 4:
                printf("End of run.\n");
                break;//terminate program
                //if invalid choice
            default:
                printf("Invalid choice.\n");
        }

        printf("\n");
    }while(c!=4);//while to the c equal to 4
}
```

The Output:

```
C:\Users\Enes\OneDrive\Documents\lab#10task1.exe

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>2
The stack is empty!

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>3
The stack is empty!

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>4
End of run.


--------------------------------
Process exited after 6.499 seconds with return value 0
Press any key to continue . . .
```

The below 2 pictures are executed at same time, they re pictures seaperated because of the not enough space.

C:\Users\Enes\OneDrive\Documents\lab#10task1.exe

```
Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>1
Enter the data to be placed on : 5
The stack in tabular form is :
AdressOfTheNode          StackValue(data)         AdressOfNextNode
007115D0                 5                        00000000

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>1
Enter the data to be placed on : 8
The stack in tabular form is :
AdressOfTheNode          StackValue(data)         AdressOfNextNode
007115E0                 8                        007115D0
007115D0                 5                        00000000

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>1
Enter the data to be placed on : 3
The stack in tabular form is :
AdressOfTheNode          StackValue(data)         AdressOfNextNode
007115F0                 3                        007115E0
007115E0                 8                        007115D0
007115D0                 5                        00000000

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>8
Invalid choice.

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>2
3 popped up.
The stack in tabular form is :
```

C:\Users\Enes\OneDrive\Documents\lab#10task1.exe

```
4 to quit.

=>8
Invalid choice.

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>2
3 popped up.
The stack in tabular form is :
AdressOfTheNode          StackValue(data)         AdressOfNextNode
007115E0                 8                        007115D0
007115D0                 5                        00000000

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>2
8 popped up.
The stack in tabular form is :
AdressOfTheNode          StackValue(data)         AdressOfNextNode
007115D0                 5                        00000000

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>3
The stack in tabular form is :
AdressOfTheNode          StackValue(data)         AdressOfNextNode
007115D0                 5                        00000000

Enter your choice
1 to place data on the stack.
2 to place data off the stack.
3 to display the stack with adress.
4 to quit.

=>4
End of run.


--------------------------------
Process exited after 28.11 seconds with return value 0
Press any key to continue . . .
```

TASK 2: Write a program that uses a stack to determine if a string is a palindrome (i.e., the string is spelled identically backward and forward). Several outputs of the program are as follows.

```
Enter a text : 123321

The text is : 123321
The reverse of the text is : 123321
The text is a polindrome.

--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

```
Enter a text : abcdcba

The text is : abcdcba
The reverse of the text is : abcdcba
The text is a polindrome.

--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

```
Enter a text : abcdef

The text is : abcdef
The reverse of the text is : fedcba
The text is a not polindrome.

--------------------------------
Process exited with return value 0
Press any key to continue . . .
```

The Code:

```
#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>

//include libraries


typedef struct snode{

        char ch;

        struct snode *next;

}nsnode;

//defining new varibale type

//with two elemts

//first keeps data second is pointing new data

//change name of variable by typedef

typedef nsnode *snptr;


//functions prototypes

int len(char *ch1);

void pushs(snptr *a,char ch1);
```

```
char pops(snptr *a);
void cmpstacks(snptr a,snptr a1);
void printstck(snptr a);
void reversestck(snptr a,snptr a1);
//this functions returns the lenght
//the string
int len(char *ch1)
{
	int i=0;

	while(*(ch1+i)!=NULL)
	   i++;

	return i;
}
//this functions push a char  to stack
void pushs(snptr *a,char ch1)
{
	//dynamic memory allocation
	snptr a1=(snode *)malloc(sizeof(snode));

	if(a1!=NULL)
	{//add char to end of the stack
		a1->ch=ch1;
		a1->next=*a;
		*a=a1;
	}

	else
	   printf("No memory available!\n");
}
```

```
//this functions delete a elemnt from stack
//delete last out element of the stack
//returns the elemnt of stack which is deleted
char pops(snptr *a)
{
	snptr a1;
	char ch1;

	a1=*a;
	ch1=(*a)->ch;
	*a=(*a)->next;
	free(a1);

	return ch1;
}
//the two functions work principle of first in last out
//and last in first out so our first value deleted lastly
void cmpstacks(snptr a,snptr a1)
{
	//this functions checking the stacks are same or not
	while(a!=NULL)
	{//checking all elemnt of the stack are same or not
		if(a->ch!=a1->ch)
		{
			printf("\nThe text is not a palindrome.\n");
			return;
		}
		a=a->next;
		a1=a1->next;
	}
	//if stacks are same display a + message
```

```
        //else disp - message
        printf("\nThe text is a palindrome\n");
}
//this functions prints stack to respect to the
//first in last out principle
void printstck(snptr a)
{
        if(a==NULL)
          printf("stack is empty!\n");

        else
          while(a!=NULL)
          {
                printf("%c",a->ch);
                a=a->next;
          }
}

main()
{
        snptr a=NULL,a1=NULL,a2=NULL,a3=NULL,a4=NULL,a5=NULL;
  char ch1[20];
  int i=0,i1;
  //defining variales
  printf("Enter a text : ");
  gets(ch1);
        //get a string from stdin
        i1=len(ch1);
        //assing length of string
        for(i=0;i<i1;i++)
        {//push char to stacks revesely because first in last out principle
```

```
            pushs(&a,ch1[i1-1-i]);
            pushs(&a1,ch1[i1-1-i]);
            pushs(&a2,tolower(ch1[i1-1-i]));
            pushs(&a3,tolower(ch1[i1-1-i]));
            //tolower because string can be uppper case and
                    //lower case at the same time
    }
    //display stacks
    printf("\nThe text is : ");
    printstck(a);

    //assign char to stacks
        //pops because it gives first element
        //the first elemnt will be the last elemnt of new stacks
    while(a1!=NULL)
        {
          pushs(&a4,pops(&a1));
      pushs(&a5,pops(&a3));
        }

        //display reverse of stack
        printf("\nThe reverse of the text : ");
        printstck(a4);

        //send function to compare stacks are same or not
        cmpstacks(a2,a5);
}
```

The Outputs:

C:\Users\Enes\OneDrive\Documents\lab#10task2.exe

```
Enter a text : 123321

The text is : 123321
The reverse of the text : 123321
The text is a palindrome

--------------------------------
Process exited after 3.359 seconds with return value 0
Press any key to continue . . .
```

C:\Users\Enes\OneDrive\Documents\lab#10task2.exe

```
Enter a text : abcdcba

The text is : abcdcba
The reverse of the text : abcdcba
The text is a palindrome

--------------------------------
Process exited after 10.53 seconds with return value 0
Press any key to continue . . .
```

C:\Users\Enes\OneDrive\Documents\lab#10task2.exe

```
Enter a text : abcdef

The text is : abcdef
The reverse of the text : fedcba
The text is not a palindrome.

--------------------------------
Process exited after 10.99 seconds with return value 0
Press any key to continue . . .
```

C:\Users\Enes\OneDrive\Documents\lab#10task2.exe

```
Enter a text : ABCdcba

The text is : ABCdcba
The reverse of the text : abcdCBA
The text is a palindrome

--------------------------------
Process exited after 11.63 seconds with return value 0
Press any key to continue . . .
```

C:\Users\Enes\OneDrive\Documents\lab#10task2.exe

```
Enter a text : ABCDef

The text is : ABCDef
The reverse of the text : feDCBA
The text is not a palindrome.

--------------------------------
Process exited after 10.08 seconds with return value 0
Press any key to continue . . .
```