EENG/INFE 212
Stacks

A stack is an ordered collection of items into which new items may be inserted and from which items may be deleted at one end called the top of the stack. A stack is a dynamic constantly changing object.
There are 2 basic operations associated with stack:

(a) *"Push" is the term used to insert an element into a stack.*
(b) *"Pop" is the term used to delete an element from the stack.*

One of the ways to implement a stack in 'C' is using a structure in the following way;

```
struct stack{
int top;
int items[STACKSIZE];
};
```

Where top is the location of the first element of the stack. If the stack is empty this value is set to -1. The array items [] contains the elements of the stack.
To "push" an element onto the stack, we need to increment top while "popping" an element from the stack decrements top.
The following is an example program for manipulating a stack.

```
#include<stdio.h>
#include<stdlib.h>
#define STACKSIZE 10
struct stack{
        int top;
        int items[STACKSIZE];
    };
int empty(struct stack *ps);
void push(struct stack *ps, int x);
int pop(struct stack *ps);
main(){
    int x,w,i;
    struct stack s;
    s.top=-1;
    /* push items into the stack */
    for(i=0;i<STACKSIZE;i++){
        scanf("%d",&x);
        push(&s,x);
    }
    /* print the items of the stack*/
    for(i=0;i<STACKSIZE;i++){
        w=pop(&s);
        printf("%d\t",w);
```

```
        }
        /* check if stack is empty*/
        if (empty(&s))  printf("stack is empty\n");
        else printf("stack is not empty\n");
}
int empty(struct stack *ps)
{
        return (ps->top == -1);
}
void push(struct stack *ps, int x)
{
        if(ps->top==STACKSIZE-1){
                printf("Stack overflow/n");
                exit(1);
        }
        else
                ps->items[++(ps->top)]=x;
}

int pop(struct stack *ps)
{
        if(empty(ps)){
                printf("Stack underflow");
                exit(1);
        }
        return(ps->items[ps->top--]);
}
```

Infix, Postfix and Prefix

Infix	Postfix	Prefix
A+B	AB+	+AB
A+B-C	AB+C-	-+ABC
(A+B)*(C-D)	AB+CD-*	*+AB-CD

This is a major application of stacks. Stacks can be used in converting from postfix to infix and vice versa and from prefix to infix and vice versa.

Lab Work

The following program accepts a postfix expression prints it out and then evaluates its value. Try to understand how it works. Compile and run the program using the following input:

6 2 3 + - 3 8 2 / + * 2 $ 3 +. What is the output?

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define MAXCOLS 80
#define TRUE 1
#define FALSE 0
```

```
struct stack{
        int top;
        int items[MAXCOLS];
    };
double eval(char[]);
int empty(struct stack *ps);
void push(struct stack *ps, int x);
double pop(struct stack *ps);
int isdigit(char);
double oper(int, double,double);
main(){
        char expr[MAXCOLS];
        int position=0;
        while((expr[position++]=getchar())!='\n')
            ;
        expr[--position]='\0';
        printf("The original postfix expression is %s",expr);
        printf("\n%f",eval(expr));
}
int empty(struct stack *ps)
{
    return (ps->top==-1);
}
void push(struct stack *ps, int x)
{
    if(ps->top==MAXCOLS-1){
        printf("Stack overflow/n");
        exit(1);
    }
    else
        ps->items[++(ps->top)]=x;
}

double pop(struct stack *ps)
{
    if(empty(ps)){
        printf("Stack underflow");
        exit(1);
    }
    return(ps->items[ps->top--]);
}
double eval(char expr[])
{
    int c,position;
    double opnd1,opnd2,value;
    struct stack opndstk;
    opndstk.top=-1;
    for(position=0;(c=expr[position])!='\0';position++)
        if(isdigit(c))
```

```
                /* operand -- convert the character representation*/
                /* of the digit into double and push it onto     */
                /*              the stack                        */
                push(&opndstk, (double) (c-'0'));
        else{
                /*        operator                               */
                opnd2=pop(&opndstk);
                opnd1=pop(&opndstk);
                value=oper(c,opnd1,opnd2);
                push(&opndstk,value);
        }


        return(pop(&opndstk));
}
int isdigit(char symb)
{
        return(symb>='0' && symb <='9');
}
double oper(int symb,double op1,double op2)
{
        switch(symb){
                case '+':return(op1+op2);
                case '-':return(op1-op2);
                case '*':return(op1*op2);
                case '/':return(op1/op2);
                case '$':return(pow(op1,op2));
                default :printf("Illegal operation");
                        exit(1);
}
}
```

HW 4.a (due after one week)

Write a C program which uses stacks and gets students ID's and midterm grades in an exam. Then the program displays ID's and midterm grades of the students in a *descending order* according *to their Midterm Grades* (i.e. starting with the student with highest grade) on the screen. In addition, the program shows the average of the students on the screen and shows the highest grade as well.

NOTES:

1) Please Send your Homework in the following Emails, but remember who is your lab instructor

emu.clab2@gmail.com for Pouya's Student
eenglab212@gmail.com for Mohamad's Student

2) Subject of email Should include student Number + Homework Number

For Example: "St. 15000012 Homework #1"

3) Your homework should be saved with your student number and attached as notepad.
For Example 15000012.txt