ASSIGNMENT#2

***Data Structures & Algorithms***

***WAQAS ASHIQ***

***BCS-F11-201***

***SECTION C***

***SUBMITTED TO: Sir Usman Ashraf***

***QUESTION#1***

***POSTFIX TO INFIX CONVERSION***

#define SIZE 50 /\* Size of Stack \*/

#include<string.h>

#include <ctype.h>

char s[SIZE];

int top=-1; /\* Global declarations \*/

push(char elem)

{ /\* Function for PUSH operation \*/

s[++top]=elem;

}

char pop()

{ /\* Function for POP operation \*/

return(s[top--]);

}

int pr(char elem)

{ /\* Function for precedence \*/

switch(elem)

{

case '#': return 0;

case ')': return 1;

case '+':

case '-': return 2;

case '\*':

case '/': return 3;

}

}

Int main(void)

{ /\* Main Program \*/

char infx[50],prfx[50],ch,elem;

int i=0,k=0;

printf("\n\nRead the Infix Expression ? ");

scanf("%s",infx);

push('#');

strrev(infx);

while( (ch=infx[i++]) != '\0')

{

if( ch == ')') push(ch);

else

if(isalnum(ch)) prfx[k++]=ch;

else

if( ch == '(')

{

while( s[top] != ')')

prfx[k++]=pop();

elem=pop(); /\* Remove ) \*/

}

else

{ /\* Operator \*/

while( pr(s[top]) >= pr(ch) )

prfx[k++]=pop();

push(ch);

}

}

while( s[top] != '#') /\* Pop from stack till empty \*/

prfx[k++]=pop();

prfx[k]='\0'; /\* Make prfx as valid string \*/

strrev(prfx);

strrev(infx);

printf("\n\nGiven Infix Expn: %s Prefix Expn: %s\n",infx,prfx);

}

***QUESTION#2***

***INFIX TO POSTFIX CONVERSION***

#include<stdio.h>

#include<conio.h>

#include<string.h>

int prece(char j);

int main(void)

{

char infix[10],stack[5],output[10],j;

int k=0,l=0,i,a,b;

stack[k]=0;

printf("write the infix expression to be converted to postfix");

scanf("%s",infix);

for(i=0;i<10;i++)

{

a=prece(infix[i]);

b=prece(stack[k]);

if(a<=b)

{

output[l]=infix[i];

l++;

}

else

{

stack[k]=infix[i];

k++;

}

}

for(i=k;i>0;i--)

{

output[l]=stack[i];

l++;

}

printf("%s",output);

getch();

}

int prece(char j)

{

if(j=='\*'||j=='/')

{

j=3;

}

else

{

if(j=='+'||j=='-')

{

j=2;

}

else

{

j=1;

}

}

return j;

}

***QUESTION#3***

***ARRANGING ELEMENTS IN STACK IN ASSENDING ORDER***

#include<stdio.h>

#define SIZE 5

int pop(int a[SIZE]);

void push(int val1,int b[SIZE],int c[SIZE]);

void display(int a[SIZE],int b[SIZE],int c[SIZE]);

int top1,top2,top3;

int a[SIZE],b[SIZE],c[SIZE];

int val1,s;

int main(){

int d=0;

printf("\n->The Size Of Array Is 5.How Many Values You Want To Enter?\n");

scanf("%d",&s);

if(s>=0 && s<=5){

printf("->Enter The Values,You Want In The First Array\n\n");

for(int i=0;i<s;i++){

printf("Enter Value %d: ",(i+1));

scanf("%d",&a[i]);

} // end for loop

top1=s-1;

top2=0,top3=0;

pop(a);

b[top2]=val1;

for(int j=s-1;j>0;j--)

{

d=pop(a);

push(d,b,c);

} // end for loop

display(a,b,c);

} // end if

else

{

printf("\n-Stack Overflow!\n");

}

} // end main function

int pop(int a[SIZE])

{

if(top1<=SIZE)

{

val1=a[top1];

a[top1]=0;

top1--;

} // end if

else

{

printf("\n-QUEUE EMPTY!!!\n");

} // end else

return val1;

} // end pop function

void push(int val1,int b[SIZE],int c[SIZE])

{

if(top2<=SIZE)

{

if(val1>b[top2])

{

for(int j=top2;j>=0;j--)

{

if(val1>b[top2])

{

c[top3]=b[top2];

b[top2]=0;

top3++;

top2--;

} // end if

} // end for loop

if(top2<0)

{

top2++;

b[top2]=val1;

}

if(val1<b[top2])

{

top2++;

b[top2]=val1;

}

for(int k=top3-1;k>=0;k--)

{

top2++;

b[top2]=c[k];

c[k]=0;

}

top3=0;

} // end if

else

{

top2++;

b[top2]=val1;

} // end else

} // end if

else

{

printf("\n-Stack Full!!!\n");

}

} // end push function

void display(int a[SIZE],int b[SIZE],int c[SIZE])

{

printf("\nArray 1 And 3 Are Now Empty.Array 2 Has Been Arranged In Ascending Order\n");

printf("\nArray 1\t\t\t Array 2\t\t Array 3\n");

for(int i=SIZE-1;i>=0;i--)

{

printf("%5d ",a[i]);

printf("%25d ",b[i]);

printf("%25d ",c[i]);

printf("\n");

}

// end for loop

printf("\n");

} // end display functi