Day 3

Infix to Postfix Coversion

CODE:

#include<stdio.h>

#include<ctype.h>

#include<string.h>

char stk[20];

int top=-1;

void push(char c)

{

stk[++top]=c;

}

char pop()

{

return(stk[top--]);

}

int priority(char c)

{

if(c=='^'|| c=='&' || c=='|')

return 3;

else if (c=='/'|| c=='\*' || c=='%')

return 2;

else if(c=='+' || c=='-')

return 1;

else

return 0;

}

main()

{

char in[50],post[50],ch;

int i,j,l;

printf("Enter the string :");

gets(in);

l=strlen(in);

j=0;

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

{

if(isalpha(in[i]))

post[j++]=in[i];

else

{

if(in[i]=='(')

push(in[i]);

else if(in[i]==')')

while((ch=pop())!='(')

post[j++]=ch;

else

{

while(priority(in[i])<=priority(stk[top]))

post[j++]=pop();

push(in[i]);

}

}

}

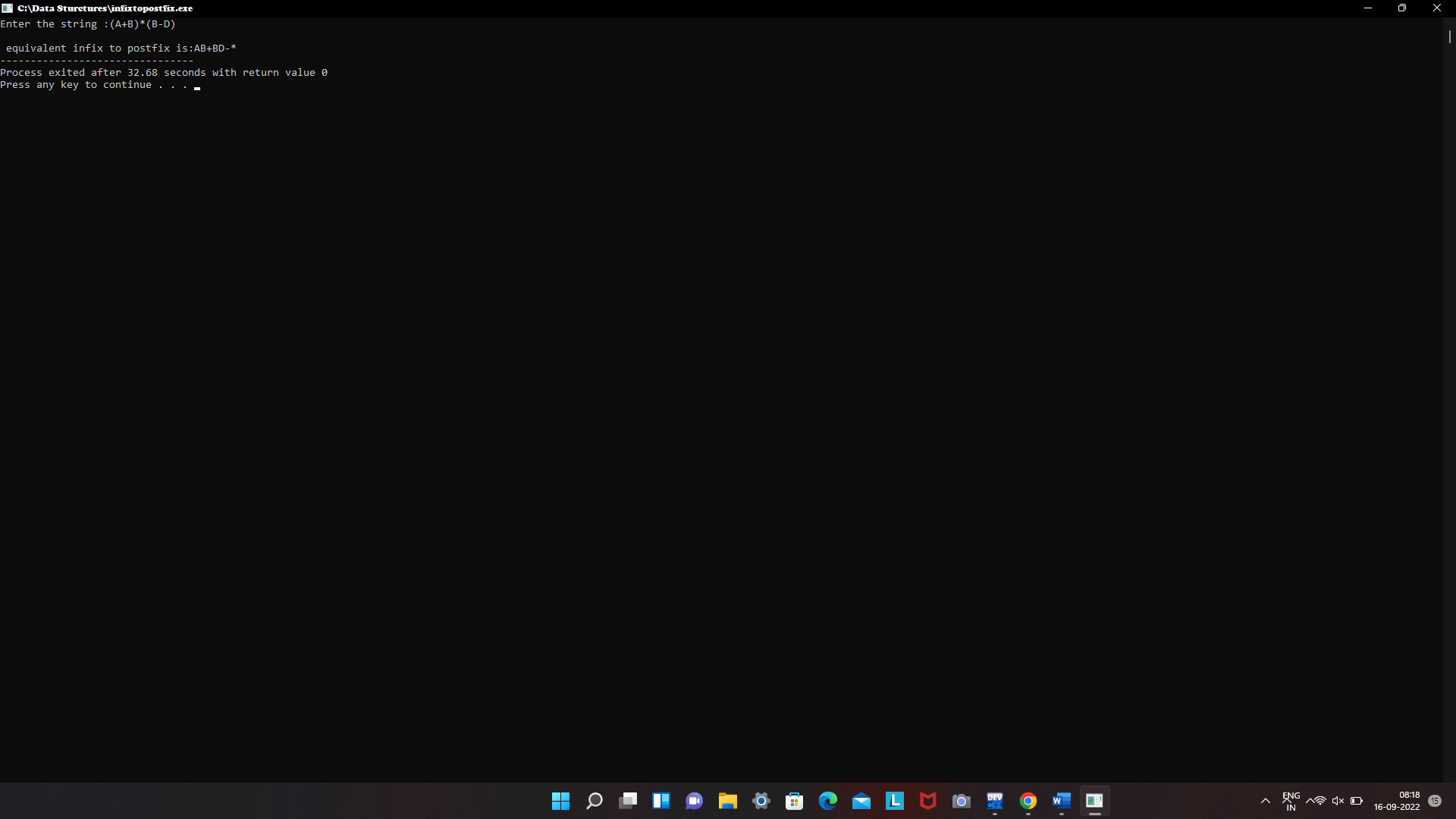
while(top!=-1)

post[j++]=pop();

post[j]='\0';

printf("\n equivalent infix to postfix is:%s",post);

}



3.Insertion Sort

Code:

#include<stdio.h>

void main( )

{

int a[20],i,j,n, temp;

printf("enter the no of elements");

scanf("%d", &n);

printf("enter the elements ");

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

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

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

{

j=i;

while (j>0&& a[j]<a[j-1])

{

temp=a[j];

a[j]=a[j-1];

a[j-1]=temp;

j--;

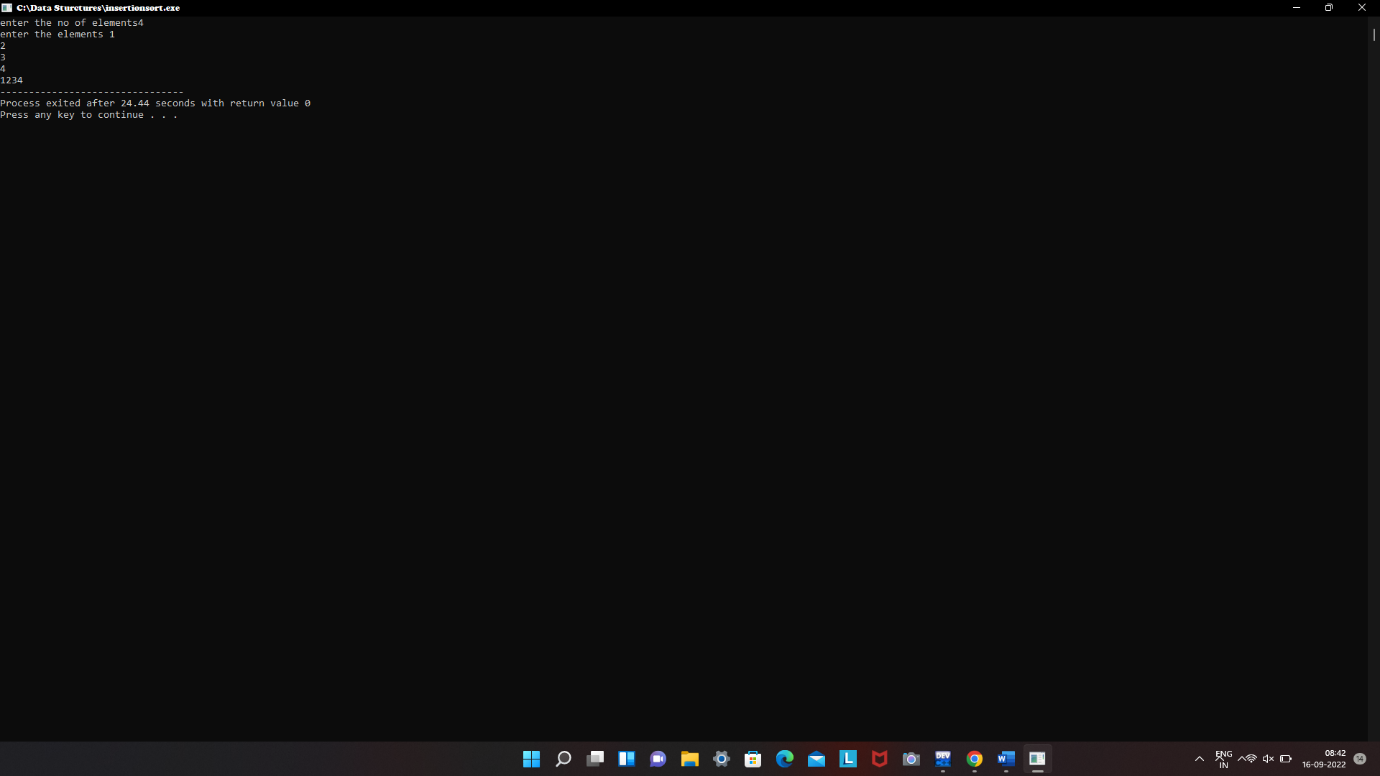
}

}

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

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

}



2.Single Linked List

Code:

**#include <stdio.h>**

**#include <malloc.h>**

**#include <stdlib.h>**

struct node {

int value;

struct node \*next;

};

void **insert**();

void **display**();

void **delete**();

int **count**();

typedef struct node DATA\_NODE;

DATA\_NODE \*head\_node, \*first\_node, \*temp\_node = 0, \*prev\_node, next\_node;

int data;

int **main**() {

int option = 0;

printf("Singly Linked List Example - All Operations\n");

while (option < 5) {

printf("\nOptions\n");

printf("1 : Insert into Linked List \n");

printf("2 : Delete from Linked List \n");

printf("3 : Display Linked List\n");

printf("4 : Count Linked List\n");

printf("Others : Exit()\n");

printf("Enter your option:");

scanf("%d", &option);

switch (option) {

case 1:

insert();

break;

case 2:

delete();

break;

case 3:

display();

break;

case 4:

count();

break;

default:

break;

}

}

return 0;

}

void **insert**() {

printf("\nEnter Element for Insert Linked List : \n");

scanf("%d", &data);

temp\_node = (DATA\_NODE \*) malloc(sizeof (DATA\_NODE));

temp\_node->value = data;

if (first\_node == 0) {

first\_node = temp\_node;

} else {

head\_node->next = temp\_node;

}

temp\_node->next = 0;

head\_node = temp\_node;

fflush(stdin);

}

void **delete**() {

int countvalue, pos, i = 0;

countvalue = count();

temp\_node = first\_node;

printf("\nDisplay Linked List : \n");

printf("\nEnter Position for Delete Element : \n");

scanf("%d", &pos);

if (pos > 0 && pos <= countvalue) {

if (pos == 1) {

temp\_node = temp\_node -> next;

first\_node = temp\_node;

printf("\nDeleted Successfully \n\n");

} else {

while (temp\_node != 0) {

if (i == (pos - 1)) {

prev\_node->next = temp\_node->next;

if(i == (countvalue - 1))

{

head\_node = prev\_node;

}

printf("\nDeleted Successfully \n\n");

break;

} else {

i++;

prev\_node = temp\_node;

temp\_node = temp\_node -> next;

}

}

}

} else

printf("\nInvalid Position \n\n");

}

void **display**() {

int count = 0;

temp\_node = first\_node;

printf("\nDisplay Linked List : \n");

while (temp\_node != 0) {

printf("# %d # ", temp\_node->value);

count++;

temp\_node = temp\_node -> next;

}

printf("\nNo Of Items In Linked List : %d\n", count);

}

int **count**() {

int count = 0;

temp\_node = first\_node;

while (temp\_node != 0) {

count++;

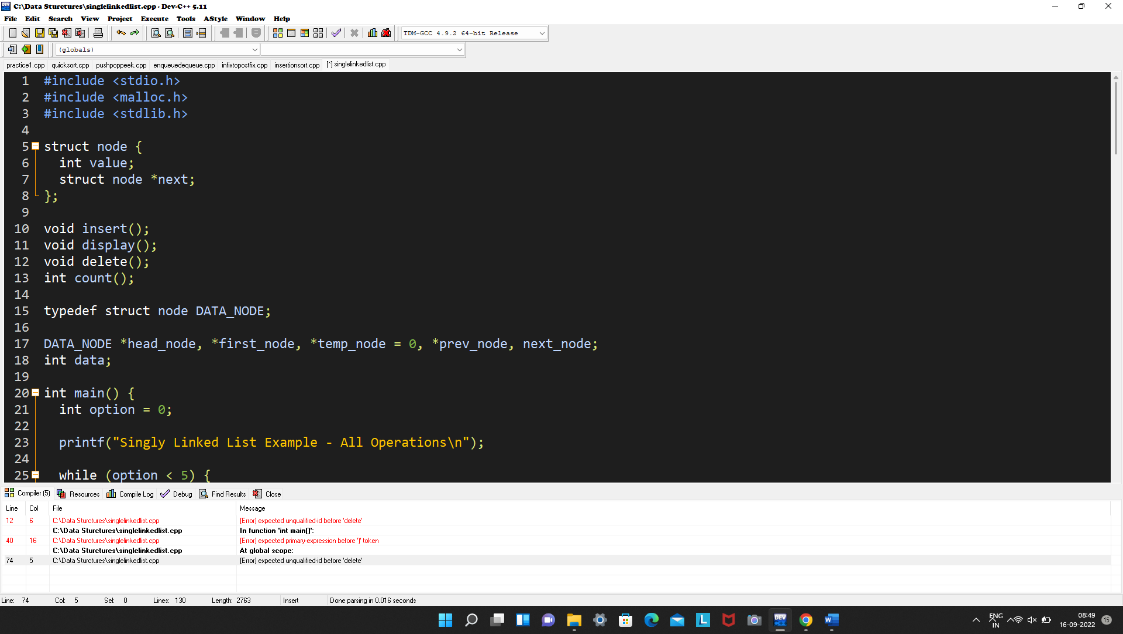
temp\_node = temp\_node -> next;

}

printf("\nNo Of Items In Linked List : %d\n", count);

return count;

}



4.Heap Sort

Code:

#include<stdio.h>

heapify(int a[], int n, int i)

{

int root,l,r,t;

root = i;

l = 2\*i + 1;

r = 2\*i + 2;

if (l<n && a[l] > a[root])

root = l;

if (r<n && a[r] > a[root])

root = r;

if (root != i)

{

t=a[i];

a[i] = a[root];

a[root] = t;

heapify(a, n, root);

}

}

heapsort(int a[], int n)

{

int i,t;

for(i=n/2-1; i>=0; i--)

heapify(a, n, i);

for(i=n-1; i>=0; i--)

{

t= a[0];

a[0]= a[i];

a[i] = t;

heapify(a, i, 0);

}

}

int main()

{

int a[50],i,n;

printf("Enter total number of elements:");

scanf("%d", &n);

printf("Enter the elements:\n");

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

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

heapsort(a,n);

printf("\n\nAfter Heap sort:\n");

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

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

return 0;

}

