13-10-2021

Pila estática

Practica: 4

Materia: Estructura de datos

Sección: D01.

Código: 216584703

Carrera: Ingeniería en computación.

Nombre alumno: Padilla Pérez Jorge Daray

Nombre profesor: Julio Esteban Valdes Lopez





Introducción

La verdad no he podido acabarla me falta todavía pensar bien como funcionaria, además de que no se exactamente que se tiene que hacer, si se tiene que hacer lo de torres de hanoi, lo que cambiar las notaciones o ambas etc.

Pantallazos

Me falta todavia falla esperare a que pueda funcionar porque todavia no entiendo muy bien para explicar los pantallazos, nomas porque se sube hoy la practica pero espero acabarlo en los siguientes dias.

Conclusión

Me falta acabarla todavía no funciona.

**Código fuente**

**#include <iostream>**

**#include <string.h>**

**#define TAMMAX 500**

**using namespace std;**

**void menu();**

**typedef char tipo\_dato;**

**struct pila{**

**tipo\_dato datos[TAMMAX];**

**void inicializa();**

**bool vacia();**

**bool llena();**

**int tope;**

**void push(tipo\_dato elem, int tope);**

**void pop(int tope);**

**int top(int tope);**

**char infijo\_posfijo(char ei[]);**

**int jerarquia\_operadores();**

**bool Esoperador();**

**//void imprimir\_torres(tipo\_dato numd);**

**//void hanoi(struct pila,struct pilaDest,struct pilaAux,int numdiscos);**

**pila(){**

**inicializa();**

**}**

**};**

**void pila::inicializa(){**

**tope = -1;**

**}**

**bool pila::vacia(){**

**if (tope == -1)**

**{**

**return true;**

**}**

**else{**

**return false;**

**}**

**}**

**bool pila::llena(){**

**if (tope == TAMMAX -1 ){**

**return true;**

**}**

**else{**

**return false;**

**}**

**}**

**//Ingresar los discos a la torre**

**void pila::push(tipo\_dato elem, int tope){**

**if (llena()){**

**cout<<"desbordamiento de datos"<<endl;**

**system("pause");**

**return;**

**}**

**else{**

**tope = tope + 1;**

**datos[tope] = elem;**

**}**

**}**

**//Sacar los discos a la torre**

**void pila::pop(int tope){**

**if (vacia()){**

**cout<<"Insuficiencia de datos"<<endl;**

**system("pause");**

**return;**

**}**

**else{**

**tope = tope - 1;**

**}**

**}**

**int pila::top(int tope){**

**if (vacia()){**

**cout<<"Insuficiencia de datos"<<endl;**

**system("pause");**

**return false;**

**}**

**else{**

**return datos[tope];**

**}**

**}**

**int pila::jerarquia\_operadores(){**

**if (datos == "+"||datos == "-"){**

**return 1;**

**}**

**if (datos == "\*"||datos == "/"){**

**return 2;**

**}**

**if (datos == "^"){**

**return 3;**

**}**

**else{**

**return 0;**

**}**

**}**

**bool pila::Esoperador(){**

**if (datos=="+" || datos=="-" || datos=="\*" || datos=="/" || datos=="^" || datos=="(" || datos==")"){**

**return true;**

**}**

**else{**

**return false;**

**}**

**}**

**char pila::infijo\_posfijo(char ei[]){**

**char ep[TAMMAX];**

**struct pila mi\_pila;**

**int i = 0,j=0;**

**while(ei[i] != '\0')**

**{**

**if(ei[i] >= 'a' && ei[i] <= 'z' || ei[i]>='A' && ei[i] <= 'Z')**

**{**

**ep[j] = ei[i];**

**i++;**

**j++;**

**}**

**else**

**if(ei[i]=='(')**

**{**

**mi\_pila.push(ei[i], tope);**

**i++;**

**}**

**else**

**if(ei[i]==')')**

**{**

**while( mi\_pila.top(tope) != '(')**

**{**

**ep[j]= tope;**

**j++;**

**}**

**if(mi\_pila.top(tope) == '(')**

**{**

**mi\_pila.pop(tope);**

**}**

**i++;**

**}**

**else{**

**if(ei[i]=='+' || ei[i]=='-' || ei[i]=='\*' || ei[i]=='/' || datos=="^")**

**{**

**if(ei[i]=='+' || ei[i] == '-')**

**{**

**while(!vacia() && mi\_pila.top(tope) != '(')**

**{**

**ep[j]=tope;**

**j++;**

**}**

**mi\_pila.push(ei[i],tope);**

**i++;**

**}**

**else if (ei[i]=='\*' || ei[i]=='/')**

**{**

**while(!vacia() && mi\_pila.top(tope) != '(' && (mi\_pila.top(tope)=='\*'|| mi\_pila.top(tope)=='/'))**

**{**

**ep[j]=(tope);**

**j++;**

**}**

**mi\_pila.push(ei[i],tope);**

**i++;**

**}**

**else**

**{**

**if(ei[i]=='^')**

**{**

**while(!vacia() && mi\_pila.top(tope) != '(' )**

**{**

**ep[j]=(tope);**

**j++;**

**}**

**mi\_pila.push(ei[i],tope);**

**i++;**

**}**

**}**

**}**

**}**

**}**

**while(!vacia())**

**{**

**ep[j]= tope;**

**j++;**

**}**

**return ep[j];**

**}**

**/\*void pila::hanoi(struct pila,struct pilaDest,struct pilaAux,int numdiscos){**

**if (numdiscos == 0){**

**return;**

**}**

**else{**

**hanoi (pilaOrig, pilaAux, pilaDest, numdiscos - 1);**

**push (top (pilaOrig), pilaDest);**

**pop (pilaOrig);**

**hanoi (pilaAux, pilaDest, pilaOrig, numdiscos - 1);**

**}**

**}**

**void pila::imprimir\_torres(tipo\_dato numd){**

**int i,j;**

**for (i=1;i<=numd;i++){**

**for (j=1;j<=numd-i;j++){**

**cout<<" ";**

**}**

**for(j=1;j<=2\*i-1;j++){**

**cout<<"\*";**

**}cout<<endl;**

**}**

**}\*/**

**struct pila mi\_pila;**

**int main()**

**{**

**int opc=0;**

**do{**

**system("cls");**

**menu();**

**cout<<"Continuar 1 salir 2:"<<endl;cin>>opc;**

**}while(opc!=2);**

**system("pause>>cls");**

**return 0;**

**}**

**void menu(){**

**char expresion\_infix[TAMMAX];**

**char expresion\_postfix[TAMMAX];**

**cout << " Longitud maxima de la expresion: "<< TAMMAX -1 << "\n" << endl;**

**cout << "\nEscribe tu expresion en infix:" << endl;**

**cin>> expresion\_infix;**

**mi\_pila.infijo\_posfijo(expresion\_infix); //**

**cout << "\nLa cadena infija \n" << expresion\_infix << endl;**

**cout << "\nConvertida en postfix es: \n" << expresion\_postfix << endl;**

**/\* int numdiscos;**

**cout<<"\*\*\*\*\*\*\tTORRE DE HANOI\t\*\*\*\*\*\*"<<endl;**

**cout<<"\nIngrese la cantidad de disco(s): "<<endl;cin>>numdiscos;**

**if (numdiscos<=0){**

**cout<<"Ingrese un numero entero positivo arriba de 0"<<endl;**

**system("pause");**

**return;**

**}**

**mi\_pila.imprimir\_torres(numdiscos);\*/**

**cout<<("\n\t");**

**system("pause");**

**}**