

Laboratorio 11

Punteros, POO y Pilas

Nombre: Gabriel Fernando Rodriguez Cutimbo

CUI: 20212157

Grupo: B

Repositorio GitHub:

https://github.com/gaco123/EPCC_CCII.git

1. Ejercicios

Resolver los siguientes ejercicios planteados:

1. Defina una Pila que permita insertar elementos utilizando clases.
2. Sobre el ejercicio anterior, adecue el programa para eliminar elementos de una Pila.
3. Implemente un algoritmo para buscar elementos de la Pila.
4. Escribir un programa que dé la solución al problema de las Torres de Hanoi para N discos, utilizando pilas, las cuales representen cada uno de los postes:



Código “Nodo.h”:

```
#ifndef NODO_H
#define NODO_H
#include <cstdlib>

class Nodo {
public:
    Nodo();
    Nodo* sig;
    int val;
private:
};

#endif
```

Código “Nodo.cpp”:

```
#include "Nodo.h"
```

```
Nodo::Nodo() {  
    sig=NULL;  
    val={};  
}
```

Código “Lab_11.cpp”:

```
/*  
    1. Defina una Pila que permita insertar elementos utilizando clases.  
    2. Sobre el ejercicio anterior, adecue el programa para eliminar elementos de una  
       Pila.  
    3. Implemente un algoritmo para buscar elementos de la Pila.  
    4. Escribir un programa que dé la solución al problema de las Torres de Hanoi para  
       N discos, utilizando pilas, las cuales representen cada uno de los postes:  
*/
```

```
#include<iostream>  
#include <locale>//Caracteres Español  
#include "Nodo.h"  
using namespace std;
```

```
//Añadir elemento al inicio de la pila
```

```
void push(Nodo*& Pila, int val){  
    Nodo* temp = new Nodo;  
    temp->val = val;  
    temp->sig = Pila;  
    Pila = temp;  
}
```

```
//Sacar elemento del final de la pila
```

```
Nodo* pop(Nodo*& Pila){  
    Nodo* temp = Pila;  
    Pila=Pila->sig;  
    temp->sig=NULL;
```

```
    return temp;
```

```
}  
Nodo* searchl(Nodo*& Pila, int val){  
    Nodo* temp = Pila;  
    while(temp->val!=val){  
        temp=temp->sig;  
    }  
    return temp;  
}
```

```
void print(Nodo*& Pila){  
    Nodo* temp = Pila;  
    if(temp!=NULL){  
        cout<<"[";  
        while(temp->sig!=NULL){  
            cout<<temp->val<<" ";  
            temp=temp->sig;  
        }  
    }
```

```
        cout<<temp->val<<"]\n";
    }
    else{
        cout<<"NULL\n";
    }
}

void T_Hanoi(int discos, Nodo*& Torre1, Nodo*& Torre2, Nodo*& Torre3, char id1, char id2, char id3,
Nodo*& TPorre1, Nodo*& TPorre2, Nodo*& TPorre3){
    Nodo* temp = NULL;
    if(discos==1){
        temp=pop(Torre1);
        cout<<"Mover disco con valor "<<temp->val<<" de la torre "<<id1<<" hacia la torre "<<id3<<endl;
        push(Torre3,temp->val);
        print(TPorre1);
        print(TPorre2);
        print(TPorre3);
        cout<<endl;
        delete temp;
        temp=NULL;
    }
    else{
        T_Hanoi(discos-1, Torre1, Torre3, Torre2, id1, id3, id2, TPorre1, TPorre2, TPorre3);
        temp=pop(Torre1);
        cout<<"Mover disco con valor "<<temp->val<<" de la torre "<<id1<<" hacia la torre "<<id3<<endl;
        push(Torre3,temp->val);
        print(TPorre1);
        print(TPorre2);
        print(TPorre3);
        cout<<endl;
        T_Hanoi(discos-1, Torre2, Torre1, Torre3, id2, id1, id3, TPorre1, TPorre2, TPorre3);
        delete temp;
        temp=NULL;
    }
}

int main (int argc, char *argv[]) {
    setlocale(LC_CTYPE, "Spanish");//Caracteres Español

    int discos;
    char id1='1';
    Nodo* Torre1 = NULL;
    char id2='2';
    Nodo* Torre2 = NULL;
    char id3='3';
    Nodo* Torre3 = NULL;

    cout<<"¿Con cuantos discos va a jugar?: ";
    cin>>discos;
    cout<<endl;
    for(int i=discos; i>=1; i--){
        push(Torre1,i);
    }
    cout<<"TORRE 1: ";
    print(Torre1);
```

```
cout<<endl;
T_Hanoi(discos,Torre1,Torre2,Torre3,id1,id2,id3,Torre1,Torre2,Torre3);
cout<<"TORRE 3: ";
print(Torre3);

return 0;
}
```

Funcionamiento:

```
{Con cuantos discos va a jugar?: 3

TORRE 1: [1, 2, 3]

Mover disco con valor 1 de la torre 1 hacia la torre 3
[2, 3]
NULL
[1]

Mover disco con valor 2 de la torre 1 hacia la torre 2
[3]
[2]
[1]

Mover disco con valor 1 de la torre 3 hacia la torre 2
[3]
[1, 2]
NULL

Mover disco con valor 3 de la torre 1 hacia la torre 3
NULL
[1, 2]
[3]

Mover disco con valor 1 de la torre 2 hacia la torre 1
[1]
[2]
[3]

Mover disco con valor 2 de la torre 2 hacia la torre 3
[1]
NULL
[2, 3]

Mover disco con valor 1 de la torre 1 hacia la torre 3
NULL
NULL
[1, 2, 3]

TORRE 3: [1, 2, 3]
```

2. Entregables

Al final estudiante deberá:

- 2.1.** Compactar el código elaborado y subirlo al aula virtual de trabajo. Agregue sus datos personales como comentario en cada archivo de código elaborado.
- 2.2.** Elaborar un documento que incluya tanto el código como capturas de pantalla de la ejecución del programa. Este documento debe estar en formato PDF.
- 2.3.** El nombre del archivo (comprimido como el documento PDF), será su LAB11_GRUPO_A/B/C_CUI_1erNOMBRE_1erAPELLIDO.
(Ejemplo: LAB11_GRUPO_A_2022123_PEDRO_VASQUEZ).
- 2.4.** Debe remitir el documento ejecutable con el siguiente formato:
LAB11_GRUPO_A/B/C_CUI_EJECUTABLE_1erNOMBRE_1erAPELLIDO
(Ejemplo: LAB11_GRUPO_A_EJECUTABLE_2022123_PEDRO_VASQUEZ).

En caso de encontrarse trabajos similares, los alumnos involucrados no tendrán evaluación y serán sujetos a sanción.