**Listas, Pilas y Colas**

Estructuras de Datos Avanzadas

Universidad Nacional de San Agustín

Nombre: Alexander Rusvell Apaza Torres

1. Con Arrays
   1. Clase VStructure: Esta Clase será padre de las clases VLista, VPila y VCola.
      1. Código:

template <int N>

class VStructure

{

protected:

int array[N];

int actual;

public:

VStructure(){

this->actual=0;

for(int i = 0; i < N; i++)

{

this->array[i]=0;

}

}

void push(int v){

if (this->actual<N)

{

this->array[actual]=v;

this->actual++;

}

}

void show(){

cout<<"Lista: | ";

for(int i = 0; i < N; i++)

{

cout<<this->array[i]<<" | ";

}

cout<<""<<endl;

}

void clear(){

for(int i = 0; i < N; i++)

{

this->array[i]=0;

}

this->actual=0;

}

bool isEmpty(){

if (actual==N)

return false;

return true;

}

bool isFull(){

return !this->isEmpty();

}

~VStructure(){

}

};

* 1. VLista:
     1. Código:

#include "VStructure.h"

template <int N>

class VLista : public VStructure<N>

{

public:

VLista(){}

~VLista(){}

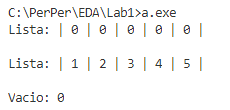
};

* + 1. Demostración:

int main()

{

VLista<5> a;

a.show();

a.push(1);

a.push(2);

a.push(3);

a.push(4);

a.push(5);

a.push(4);

cout<<endl;

a.show();

cout<<endl<<"Vacio: "<< a.isEmpty()<<endl;

return 0;

}

* 1. VPila
     1. Código

template <int N>

class VPila : public VStructure<N>

{

public:

VPila(){}

~VPila(){}

int pop(){

int res;

this->actual--;

res = this->array[this->actual];

this->array[this->actual]=0;

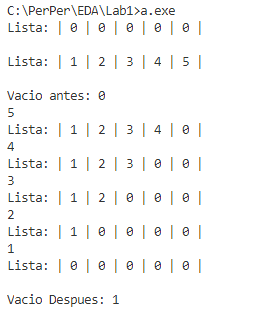
return res;

}

};

* + 1. Demostración

int main()

{

VPila<5> a;

a.show();

a.push(1);

a.push(2);

a.push(3);

a.push(4);

a.push(5);

a.push(4);

cout<<endl;

a.show();

cout<<endl<<"Vacio antes: "<< a.isEmpty()<<endl;

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<endl<<"Vacio Despues: "<< a.isEmpty()<<endl;

return 0;

}

* 1. VCola
     1. Código

template<int N>

class VCola : public VStructure<N>

{

public:

VCola(){}

int pop()

{

if (this->actual==0)

return -1;

int res= this->array[0];

for(int i = 1; i < this->actual ; i++){

this->array[i-1]=this->array[i];

}

this->array[this->actual-1]=0;

this->actual--;

return res;

}

~VCola(){

}

};

* + 1. Demostración

int main()

{

VCola<5> a;

a.show();

a.push(1);

a.push(2);

a.push(3);

a.push(4);

a.push(5);

a.push(4);

cout<<endl;

a.show();

cout<<endl<<"Vacio antes: "<< a.isEmpty()<<endl;

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<a.pop()<<endl;

a.show();

cout<<endl<<"Vacio Despues: "<< a.isEmpty()<<endl;

return 0;

}

1. Con Punteros
   1. PStructure: Esta clase será padre de las clases PLista, PCola y PPila
      1. Código
         1. Clase PNodo

class PNodo

{

private:

int value;

PNodo\* siguiente;

public:

PNodo(int );

~PNodo();

friend class PStructure;

};

PNodo::PNodo(int v)

{

this->value=v;

this->siguiente=0;

}

PNodo::~PNodo()

{

}

* + - 1. Clase PStructure

class PStructure

{

protected:

PNodo\* first;

PNodo\* last;

public:

PStructure(){

this->first=0;

this->last=0;

}

void push(int v){

PNodo\* nuevo=new PNodo(v);

if (last){

last->siguiente=nuevo;

last=nuevo;

return;

}

first=last=nuevo;

return;

}

void show()

{

PNodo\* temp= this->first;

cout<<"Lista: | ";

while(temp){

cout<<temp->value<<" | ";

temp=temp->siguiente;

}

cout<<endl;

}

void remove(int position){

PNodo\* temp=this->first;

if (position==0){

this->first=this->first->siguiente;

}

for(int i = 0; i < position-1; i++){

temp=temp->siguiente;

}

cout<<endl<<"Dato Borrado: "<<temp->value<<endl;

if (!temp){

return;

}

if (temp->siguiente && temp->siguiente != this->last){

temp->siguiente=temp->siguiente->siguiente;

}

else if(temp->siguiente && temp->siguiente==last){

temp->siguiente=0;

}

else if(!temp->siguiente){

this->first=0;

this->last=0;

}

}

~PStructure(){}

};

* 1. Clase PLista
     1. Código:

class PLista : public PStructure

{

public:

PLista(){}

~PLista(){}

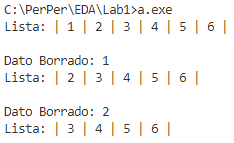
};

* + 1. Demostración:

int main()

{

PLista a;

a.push(1);

a.push(2);

a.push(3);

a.push(4);

a.push(5);

a.push(6);

a.show();

a.remove(0);

a.show();

a.remove(0);

a.show();

}

* 1. Clase PCola
     1. Código

class PCola : public PStructure

{

public:

PCola(){}

int pop()

{

PNodo\* temp=this->first;

PStructure::remove(0);

return temp->value;

}

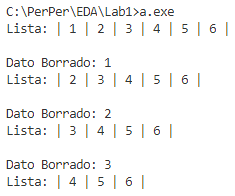
~PCola(){}

};

* + 1. Demostración

int main()

{

PCola a;

a.push(1);

a.push(2);

a.push(3);

a.push(4);

a.push(5);

a.push(6);

a.show();

a.pop();

a.show();

a.pop();

a.show();

a.pop();

a.show();

}

* 1. Clase PPila
     1. Código

class PPila : public PStructure

{

public:

PPila(){}

int pop()

{

PNodo\* temp= this->first;

while(temp->siguiente!= this->last){

temp=temp->siguiente;

}

int res=temp->siguiente->value;

temp->siguiente=0;

this->last=temp;

cout<<endl<<"Dato Borrado: "<<res<<endl;

return res;

}

~PPila(){}

};

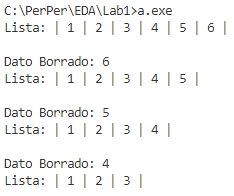
* + 1. Demostración

int main()

{

PPila a;

a.push(1);

a.push(2);

a.push(3);

a.push(4);

a.push(5);

a.push(6);

a.show();

a.pop();

a.show();

a.pop();

a.show();

a.pop();

a.show();

}