“UNIVERSIDAD NACIONAL DE SAN AGUSTÍN”

FACULTAD DE INGENIERÍA,PRODUCCIÓN Y SERVICIOS
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACIÓN

CURSO:

Ciencias de la Computación - Grupo “B”

DOCENTE:

Enzo Edir Velásquez Lobatón

ALUMNO:

Fabricio Huaquisto Quispe

REPOSITORIO:

https://github.com/fhuaquisto21/EPCC-CCII

Arequipa - Perú

2022

1. **node.h**

```
class Node {
  private:
    int value;
    Node* next;
  public:
    Node(int);
    ~Node();
    int getValue();
    Node* getNext();
    void setValue(int);
    void setNext(Node*);
};
```

2. **node.cpp**

```
#include <iostream>
#include "node.h"

Node::Node(int _value) {
  this->value = _value;
  this->next = nullptr;
}

Node::~Node() {}

Node* Node::getNext() {
  return this->next;
}

int Node::getValue() {
  return this->value;
}

void Node::setNext(Node* _next) {
  this->next = _next;
}

void Node::setValue(int _value) {
  this->value = _value;
}
```

3. **pila.h**

```
#include "node.cpp"

class Pila {
  private:
    Node* head;
    int length;
  public:
    Pila();
    Pila(int);
    ~Pila();
    int push(int);
    int pop();
    void printPila();
    int search(int);
};
```

4. **pila.cpp**

```
#include "pila.h"

Pila::Pila() {
  this->head = nullptr;
  this->length = 0;
}

Pila::Pila(int _value) {
  this->head = new Node(_value);
  this->length = 1;
}

Pila::~Pila() {}

int Pila::push(int _value) {
  Node* newNode = new Node(_value);
  if (this->head != nullptr) {
    newNode->setNext(this->head);
  }
  this->head = newNode;
  ++this->length;
  return this->head->getValue();
}

int Pila::pop() {
  if (this->head != nullptr) {
    Node* auxNode = this->head;
    int auxNodeValue = auxNode->getValue();
    this->head = this->head->getNext();
    delete auxNode;
    --this->length;
    return auxNodeValue;
  }
  return 0;
}

void Pila::printPila() {
  if (this->head == nullptr) {
    std::cerr << "ERROR: La pila está vacía";
    exit(-1);
  }
  Node* currentNode = this->head;
  while (currentNode->getNext() != nullptr) {
    std::cout << currentNode->getValue() << " -> ";
    currentNode = currentNode->getNext();
  }
  std::cout << currentNode->getValue() << std::endl;
}

int Pila::search(int _i) {
  if (_i >= this->length || _i < 0) {
    return 0;
  }
  Node* currentNode = this->head;
  for (int i = 0; i < _i; ++i) {
    currentNode = currentNode->getNext();
  }
  return currentNode->getValue();
}
```

5. **main.cpp**

```
#include <iostream>
#include "pila.cpp"
using namespace std;

void printMenu() {
  cout << "[1] Agregar nodo" << endl;
  cout << "[2] Eliminar nodo" << endl;
  cout << "[3] Buscar nodo" << endl;
  cout << "[4] Imprimir pila" << endl;
  cout << "[0] Salir" << endl;
  cout << endl << "Option: ";
}

int main() {
  Pila* pila = new Pila();
  int opt = 0;
  int value;
  do {
    printMenu();
    cin >> opt;
    printf("\e[1;1H\e[2J");
    switch (opt) {
      case 0:
        break;
      case 1:
        cout << "Valor del nuevo nodo: ";
        cin >> value;
        pila->push(value);
        break;
      case 2:
        pila->pop();
        break;
      case 3:
        cout << "ïndice del nodo a buscar: ";
        cin >> value;
        cout << "Su valor es: " << pila->search(value) << endl;
        break;
      case 4:
        pila->printPila();
        break;
    }
  } while (opt != 0);
}
```