

Curso: Algorítmica y fundamentos de programación

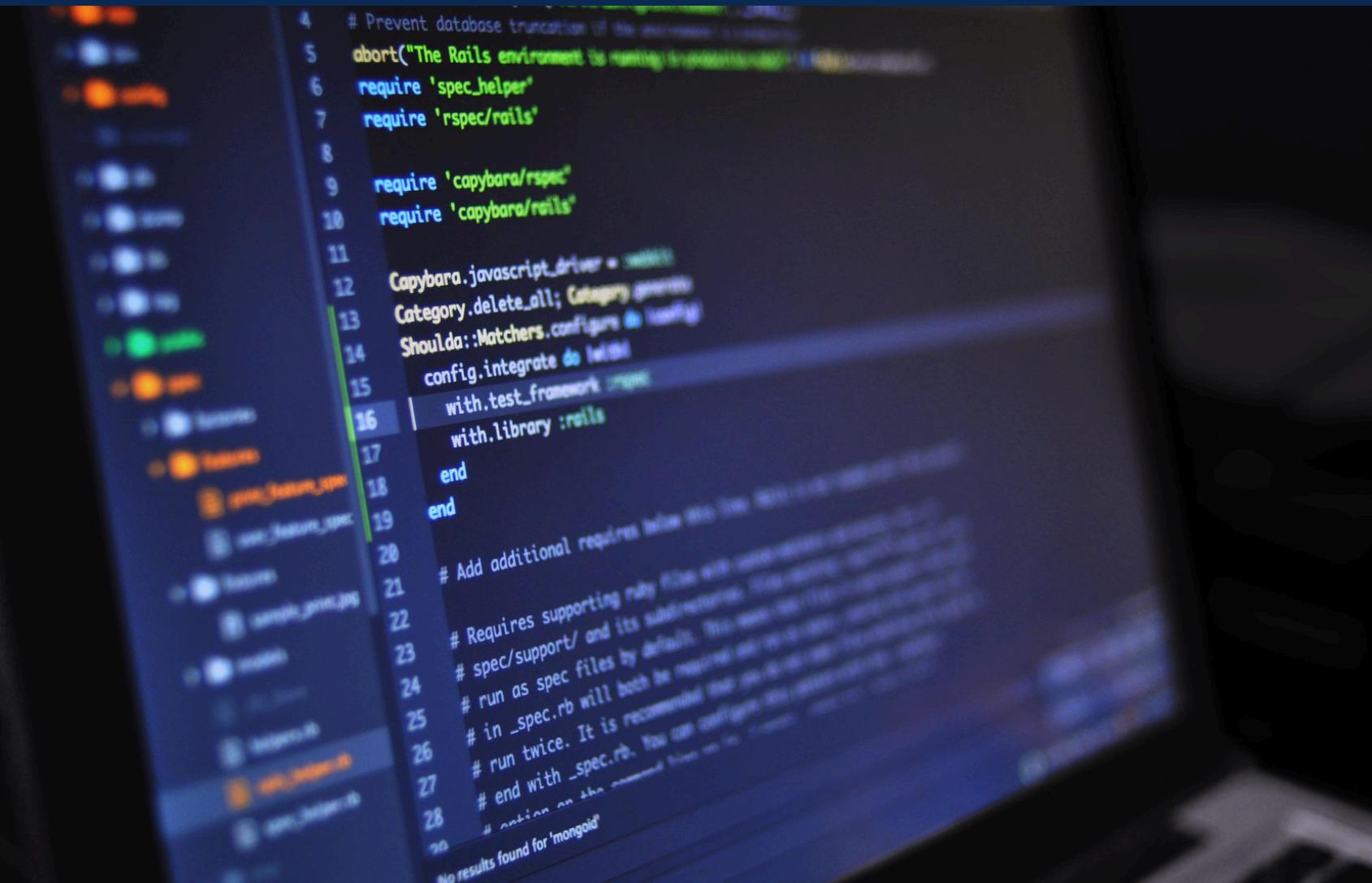


Semana 11 - Grupo 8

Integrantes:

- Ayzanoa Solano Joao Carlos
- Huamani Huaman Victor Jesus
- Salazar Espiritu Rodrigo Ahmed

1. VECTORES CON LA IMPLEMENTACIÓN DE PILA Y COLA. EJERCICIOS ADICIONALES



1.1 DEFINICION

- Pila (stack): Es una estructura de datos LIFO (Last In, First Out). Los métodos push y pop agregan y eliminan elementos respectivamente. top te permite ver el último elemento de la pila sin eliminarlo.
- Cola (queue): Es una estructura de datos FIFO (First In, First Out). Los métodos enqueue y dequeue agregan y eliminan elementos respectivamente. front devuelve el primer elemento de la cola.

EJERCICIO 1: IMPLEMENTACIÓN DE UNA PILA (STACK) EN UN VECTOR.

- Vamos a usar un vector en C++ para implementar una pila. Los elementos serán agregados al final del vector, y la eliminación será siempre desde el final (el "top").



<https://www.programiz.com/online-compiler/6CcKEk1aJpW47>

Código de Pila en C++

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 class Pila {
6 private:
7     vector<int> pila; // El vector que representa la pila
8
9 public:
10    // Aadir un elemento a la pila
11    void push(int valor) {
12        pila.push_back(valor);
13        cout << "Elemento " << valor << " agregado a la pila." <<
14            endl;
15    }
16
17    // Eliminar el elemento superior de la pila
18    void pop() {
19        if (pila.empty()) {
20            cout << "Pila vac a! No se puede hacer pop." << endl;
21        } else {
22            cout << "Elemento " << pila.back() << " eliminado de la
23                pila." << endl;
24            pila.pop_back(); // Elimina el ltimo elemento
25        }
26    }
27
28    // Mostrar el elemento superior de la pila
29    void mostrarTop() {
30        if (pila.empty()) {
31            cout << "La pila est vac a." << endl;
32        } else {
33            cout << "Elemento en el top: " << pila.back() << endl;
34        }
35    }
36
37 int main() {
38     Pila miPila;
39
40     // Aadiendo elementos a la pila
41     miPila.push(10);
42     miPila.push(20);
43     miPila.push(30);
44
45     // Mostrar el top de la pila
46     miPila.mostrarTop();
47
48     // Eliminando elementos de la pila
49     miPila.pop();
50     miPila.mostrarTop();
51
52 }
```

Output

Elemento 10 agregado a la pila.
Elemento 20 agregado a la pila.
Elemento 30 agregado a la pila.
Elemento en el top: 30
Elemento 30 eliminado de la pila.
Elemento en el top: 20

EJERCICIO 2: COLA (QUEUE) CON VECTORES.

- En este ejemplo, vamos a usar un vector para implementar una cola. Los elementos se agregarán al final del vector, y los elementos se eliminarán desde el principio del vector (el "front").



<https://www.programiz.com/online-compiler/1jRlzyGilvK3F>

```

1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 class Cola {
6 private:
7     vector<int> cola; // El vector que representa la cola
8
9 public:
10    // Aadir un elemento a la cola
11    void enqueue(int valor) {
12        cola.push_back(valor);
13        cout << "Elemento " << valor << " agregado a la cola." <<
14            endl;
15    }
16
17    // Eliminar el primer elemento de la cola
18    void dequeue() {
19        if (cola.empty()) {
20            cout << "Cola vac a! No se puede hacer dequeue." <<
21                endl;
22        } else {
23            cout << "Elemento " << cola.front() << " eliminado de
24                la cola." << endl;
25            cola.erase(cola.begin()); // Elimina el primer
26                elemento
27        }
28    }
29
30    // Mostrar el primer elemento de la cola
31    void mostrarFrente() {
32        if (cola.empty()) {
33            cout << "La cola est vac a." << endl;
34        } else {
35            cout << "Elemento en el frente: " << cola.front() <<
36                endl;
37        }
38    }
39 };
40
41 int main() {
42     Cola miCola;
43
44     // Aadiendo elementos a la cola
45     miCola.enqueue(10);
46     miCola.enqueue(20);
47     miCola.enqueue(30);
48
49     // Mostrar el primer elemento de la cola
50     miCola.mostrarFrente();
51
52     // Eliminando elementos de la cola
53     miCola.dequeue();
54     miCola.mostrarFrente();
55 }
```

Output

Elemento 10 agregado a la cola.
Elemento 20 agregado a la cola.
Elemento 30 agregado a la cola.
Elemento en el frente: 10
Elemento 10 eliminado de la cola.
Elemento en el frente: 20

EJERCICIOS FEL GRUPO 4 USANDO PILAS Y COLAS EN VECTORES

- EJERCICIO 10

10) Escribir un programa que permita a un usuario ingresar caracteres en una matriz , informar luego la matriz completa pero con la primer y última fila ordenada alfabéticamente, informar también cuantas letras “a” se ingresaron.

```
1 #include <iostream>
2 #include <queue>
3 #include <algorithm>
4 using namespace std;
5
6 int main() {
7     char matriz[4][4];
8     queue<char> cola;
9     int contadorA = 0;
10
11    // Letras ya cargadas para no escribir 16 veces
12    string letras = "fbazklmnctdavwbz";
13    for (char c : letras) {
14        if (c == 'a') contadorA++;
15        cola.push(c);
16    }
17
18    // Llenar matriz desde la cola
19    for (int i = 0; i < 4; i++) {
20        for (int j = 0; j < 4; j++) {
21            matriz[i][j] = cola.front();
22            cola.pop();
23        }
24
25    // Ordenar primera y última fila
26    sort(matriz[0], matriz[0] + 4);
27    sort(matriz[3], matriz[3] + 4);
28
29    // Mostrar matriz
30    cout << "Matriz final:\n";
31    for (int i = 0; i < 4; i++) {
32        for (int j = 0; j < 4; j++)
33            cout << matriz[i][j] << " ";
34        cout << endl;
35    }
36
37    cout << "\nCantidad de letras 'a': " << contadorA << endl;
38    return 0;
39 }
40
```

Output

Matriz final:

```
a b f z
k l m n
c t d a
b v w z
```

Cantidad de letras 'a': 2

EJERCICIOS FEL GRUPO 4 USANDO PILAS Y COLAS EN VECTORES

EJERCICIO 14

14) Se ingresan datos a un vector de enteros de 8 elementos, escribir la función `int BuscaVal(int v[], int val);` la cual recibirá el vector por referencia y la variable `val`, dicha función devolverá 1 si `val` existe en el vector, de lo contrario devolverá -1.

```
1 #include <iostream>
2 #include <stack>
3 using namespace std;
4
5 int main() {
6     stack<int> pila;
7     int val;
8
9     // Datos fijos para no ingresar uno por uno
10    int datos[] = {3, 5, 7, 1, 8, 4, 9, 2};
11    for (int i = 0; i < 8; i++) pila.push(datos[i]);
12
13    cout << "Buscar valor: ";
14    cin >> val;
15
16    while (!pila.empty()) {
17        if (pila.top() == val) {
18            cout << "Valor encontrado en la pila.\n";
19            return 0;
20        }
21        pila.pop();
22    }
23
24    cout << "Valor NO encontrado.\n";
25    return 0;
26 }
27
```

Output

```
Buscar valor: 20
Valor NO encontrado.
```

MUCHAS GRACIAS

```
    render() {
      return (
        <React.Fragment>
          <div className="py-5">
            <div className="container">
              <Title name="our" title="product">
              <div className="row">
                <ProductConsumer>
                  {(value) => {
                    |   |   |   console.log(value)
                  }}
                </ProductConsumer>
              </div>
            </div>
          </div>
        <React.Fragment>
```