Ayudantía N°5

Queue y Stack

Ayudante: Natalia Romero

Programación Avanzada S07

Queue

- Es un contenedor de objetos en el cual se ingresan y se eliminan datos con la metodología FIFO (Primero en entrar es el primero en salir).

Queue

- Debemos incluír esta librería:

```
#include <queque>
```

Queue

- Declaración:

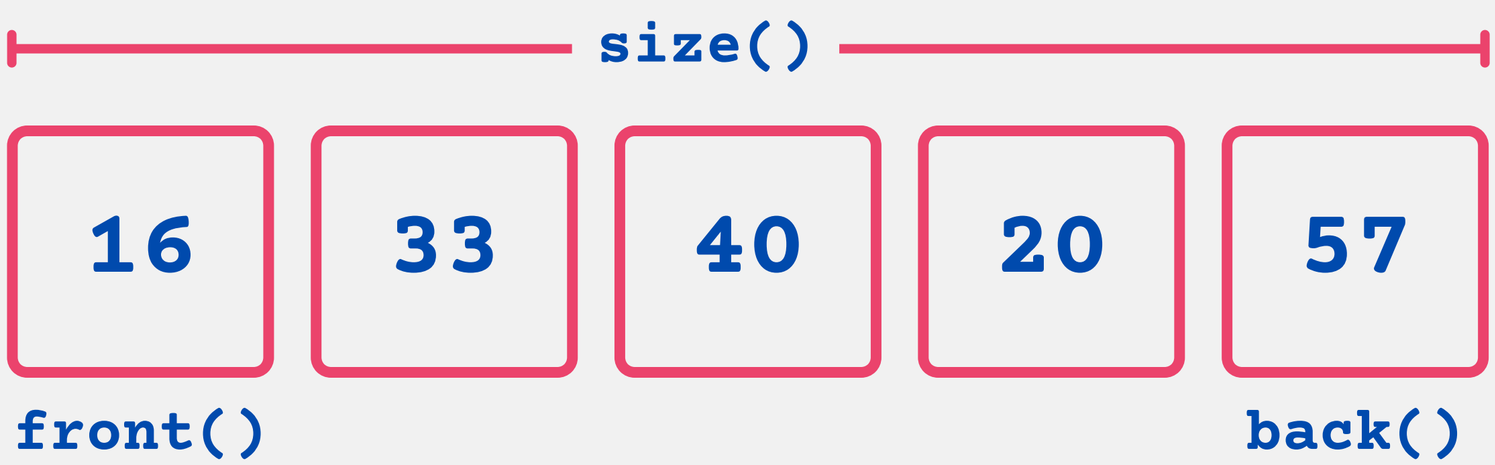
```
queue<tipo_de_dato> nombre;
```

Queue

- **push(*item*):** Inserta un nuevo elemento al final de la queue.
- **pop():** Elimina el elemento frontal de la queue.
- **front():** Devuelve el primer elemento.
- **back():** Devuelve el último elemento.
- **empty():** Devuelve true si la cola está vacía.
- **size():** Devuelve el tamaño de la cola.
- **swap(*queue*):** Intercambia valores de una cola a otra cola dada.

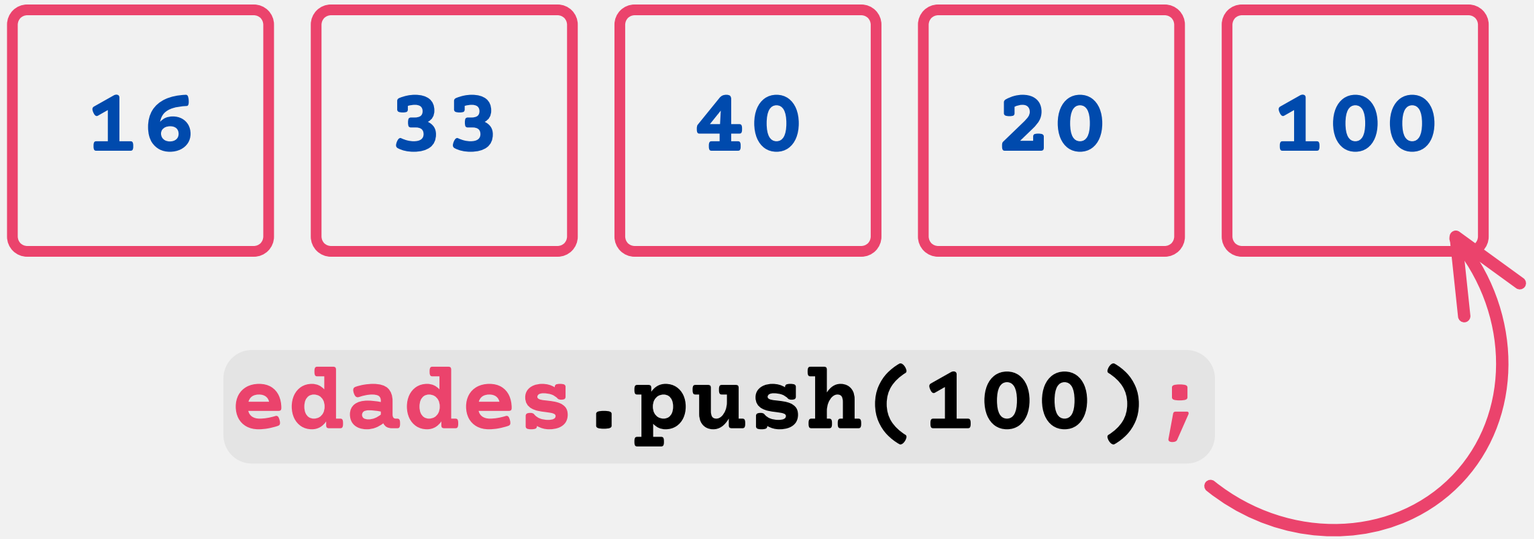
Queue

```
queue<int> edades;
```



Queue

```
queue<int> edades;
```




```
edades.push(100);
```

Queue

```
queue<int> edades;
```


```
edades.pop();
```

Ejercicio 1

- El casino de la FIC requiere un sistema de atención para la venta de tickets de almuerzo. Se tienen 500 tickets disponibles con el valor de $1.000 C/U. Normalmente se genera una fila inmensa de estudiantes, los cuales se identifican con un rut, y además cuentan con una tarjeta con cupo inicial de $30.000. Con toda esta información se pide:
  - Crear clase **Estudiante** que tenga como atributos rut y saldo. Agregar **Getters y Setters**. Para setSaldo disminuir el precio de un ticket.
  - Crear clase **Casino** que contenga una fila de estudiantes, con los siguientes metodos:
    - **imprimirFila():** Imprime la fila actual de estudiantes con su rut.
    - **ingreso(estudiante):** Recibe un estudiante y lo ingresa a la fila.
    - **stock():** Retorna true si el stock de tickets es mayor a 0.
    - **venta():** Se hace venta de ticket al primer estudiante en la fila. Primero debe verificar **stock()** y que el estudiante cuente con saldo para la venta de su ticket, si no cumple debe enviar un mensaje y se debe retirar de la fila, de lo contrario realizar la venta disminuyendo saldo del estudiante , stock de tickets y retirar.

Stack

- Es un contenedor de objetos en el cual se ingresan y se eliminan datos con la metodología LIFO (Último en entrar es el primero en salir).

Stack

- Debemos incluír esta librería:

```
#include <stack>
```

Stack

- Declaración:

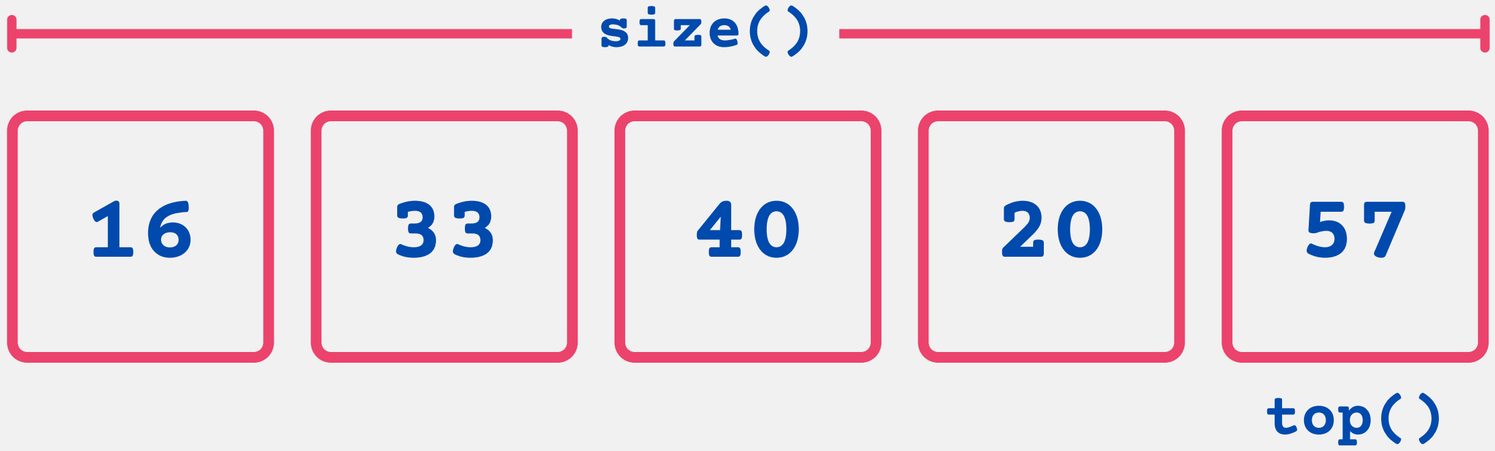
```
stack<tipo_de_dato> nombre;
```

Stack

- **push(*item*):** Inserta un nuevo elemento al final de la pila.
- **pop():** Elimina el elemento frontal de la pila.
- **top():** Devuelve el último elemento.
- **empty():** Devuelve true si la pila está vacía.
- **size():** Devuelve el tamaño de la pila.
- **swap(*stack*):** Intercambia valores de una pila a otra pila dada.

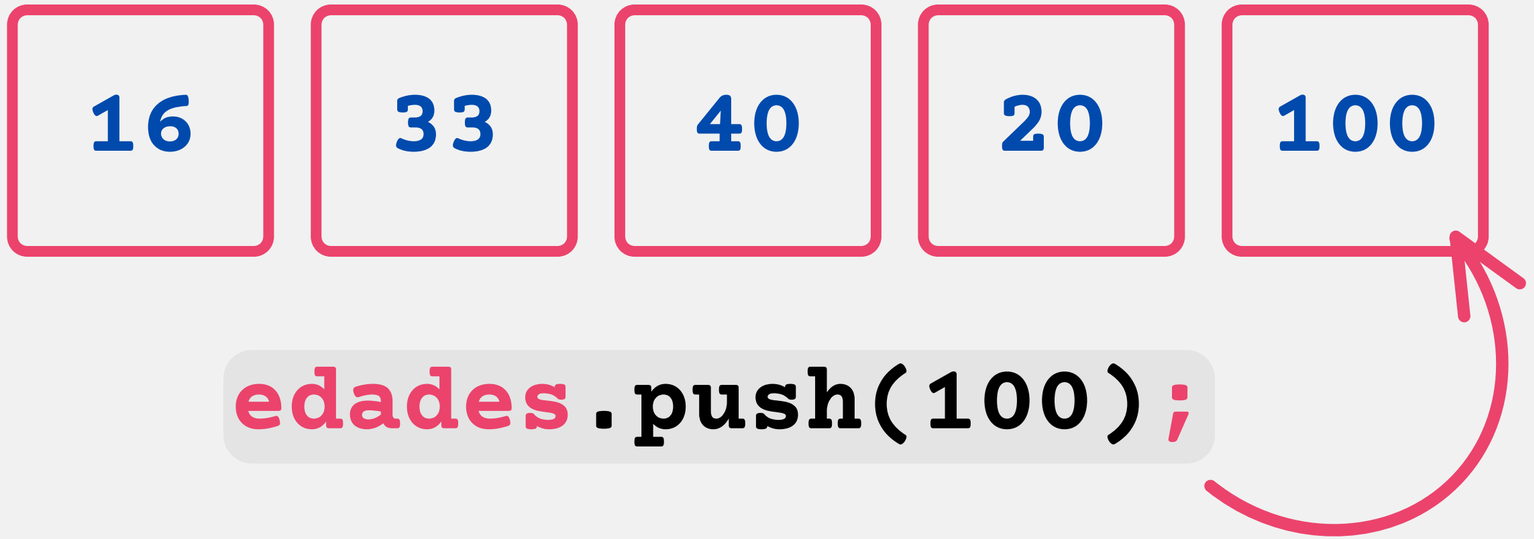
Stack

```
stack<int> edades;
```



Stack

```
stack<int> edades;
```




```
edades.push(100);
```

Stack

```
stack<int> edades;
```


```
edades.pop();
```

Ejercicio 2

- La UDPyme es una tiendita que vende discos musicales. El dueño es muy desordenado y tiene los discos apilados en una caja sin saber cual es cual.
- Se venden 3 tipos de discos: rock, reggaeton y pop. Cada disco además de diferenciarse por el tipo, tiene un nombre, cantante, año de publicación y valor.
- Se necesita que se creen diferentes torres ordenadas de discos según el tipo (3 torres en total).
- Cree un programa que permita contener la pila de discos en desorden y luego ordenarlos en 3 distintas. Se debe tener una clase Disco con sus atributos y otra UDPyme que guarde y modifique las pilas.

¿Dudas, consultas?

Contacto

+569 53890796
natalia.romero_g@mail.udp.cl

Material

https://github.com/natalia-romero/progra_av2022.git