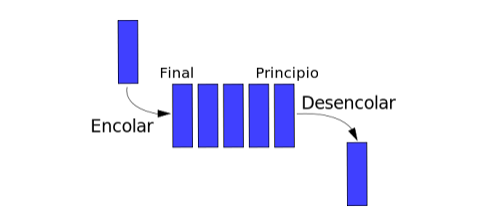
**COLAS**:

1. Explique textual y gráficamente que es un una Cola, y cuales son las clases ofrece el SDK de Java para trabajar con Colas

Las colas (también llamadas queue) son un conjunto de elementos ordenados donde el primer elemento en entrar también es el primero en salir.

Debido a esto, las colas son consideradas estructuras de datos FIFO (First in, First Out).



Tomado de: http://aprenderestructuradedatos.blogspot.com/2015/11/colas-en-java.html

Las colas poseen ciertas operaciones básicas como:

* CrearCola()
* Encolar()
* Desencolar()
* ColaVacia()
* ColaLlena()

Ciertamente, java ofrece clases como LinkedList y ArrayDeque para trabajar con la interfaz Queue (Cola); A su vez estas ofrecen funciones como add() para manejar los datos de dicha cola.

1. Explique textual y gráficamente que es un una Pila, y cuales son las clases ofrece el SDK de Java para trabajar con Pilas

Las pilas (también llamadas stack) son un conjunto de elementos ordenados donde solo se puede ingresar y eliminar elementos desde un solo extremo llamado cima (o top) que se encuentra en dicha pila. Debido a esto, las pilas son consideradas estructuras de datos LIFO (Last in, First Out).

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Tomado de: https://es.wikipedia.org/wiki/Pila\_(informática)

Las pilas poseen ciertas operaciones básicas como:

* Apilar()
* Desapilar()
* CrearPila()
* Cima()

Ciertamente java ofrece una clase llamada Stack, la cual facilita el trabajar con pilas en Java. Sus operación básicas son push(introduce un elemento), pop(saca un elemento), peek(regresa el elemento que esta en la cima), empty(comprueba si la pila esta vacia) y search(la cual busca un elemento en la pila y regresa su posicion).

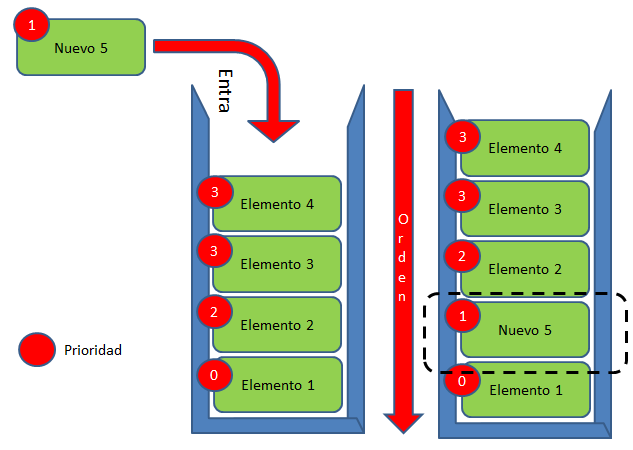
1. Explique por qué las colas y pilas se consideran elementos fundamentales en las estructuras de datos.

Estos elementos presentes en la Estructura de Datos son de vital importancia dentro de la programación debido a que propician distintas formas de organización y manejo de datos que poseen cierta relación o cierto valor. De tal manera que permiten el procesamiento eficiente de grandes cantidades de información.

1. Explique gráfica y textualmente que es una cola con prioridades

Una cola con prioridades es una cola especial de datos en los cuales sus elementos se ordenan teniendo en cuenta un orden natural u orden personalizado al momento de crear la cola.

Java ofrece una interfaz llamada PriorityQueue la cual facilita la creación de colas con prioridades



Tomado de: http://pilascolasenjavaestructuradedatos.blogspot.com/2016/05/colas-con-prioridad-en-java.html

1. Explique gráfica y textualmente que es una Cola con comportamiento LIFO

Como su nombre lo indica, LIFO “*Last In, First Out*” o “*Ultimo en Entrar, Primero en salir*”, son listas cuyos elementos son apilados y solo pueden ser ingresados y sacados por un extremo (generalmente llamado cima o top).

Un gran ejemplo de esto son las pilas, cuyo concepto y cuyos métodos para operar con datos cumplen con la definición de LIFO.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

1. Explique gráfica y textualmente que es una cola con comportamiento FIFO

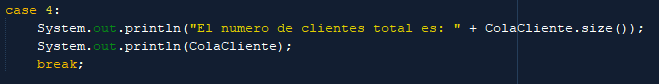
Como su nombre lo indica, FIFO “*First In, First Out*” o “*Primero en Entrar, Primero en salir*”, son listas cuyos elementos son ordenados teniendo en cuenta su orden de llegada. Dichos elementos son ingresados mediante un extremo y sacados por el extremo opuesto.

Un gran ejemplo de esto son las colas, cuyo concepto y cuyos métodos para operar con datos cumplen con la definición de FIFO.

Gráfico en cascada

Descripción generada automáticamente

1. Explique textual y gráficamente cual es el algoritmo o proceso para **contar** los elementos de una Cola, desde el primero que entra (la Cabeza) hasta el último que entra (el final). Hacer el ejemplo en Java.



Debido a que implementamos la cola usando Queue y LinkedList, esta posee métodos para determinar el numero de elementos. En este caso hacemos un print del tamaño de la cola, usando el método .size().

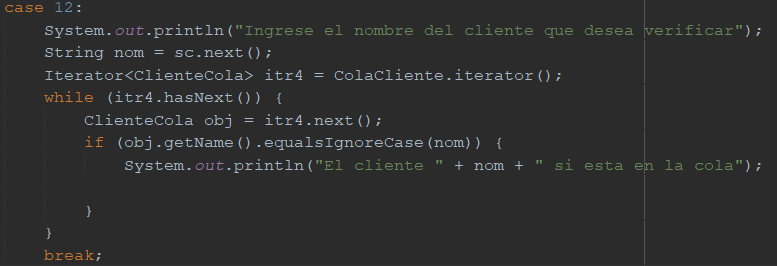
1. Explique textual y gráficamente cual es el algoritmo o proceso para **saber si una Cola está vacía o no**. Hacer un ejemplo en Java.

Texto

Descripción generada automáticamente

Debido a que implementamos la cola usando Queue y LinkedList, esta posee métodos para verificar si la cola esta vacia. En este caso usamos un if para evaluar si el método .isEmpty() regresa true o false, si este regresa true significa que esta vacia y si regresa false es porque tiene elementos.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **buscar** un elemento en una Cola. Hacer un ejemplo en Java.



En este caso, pedimos por consola que coloquen el nombre del cliente que desean verificar en la cola. Recorremos la cola mediante un Iterator (que hace parte de los métodos presentes en LinkedList) y le decimos al iterador que cuando encuentre un nombre igual al ingresado, diga que ese cliente si esta en la cola.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **agregar un elemento** a una Cola. Hacer un ejemplo en Java.

Texto

Descripción generada automáticamente

Queue y LinkedList proveen el método add() el cual añade un objeto a la Cola. La cola es de tipo ClienteCola (String, int) por tanto para añadir un cliente pedimos el nombre por teclado y su índice es el incremento del indice guardado en el setIndice().

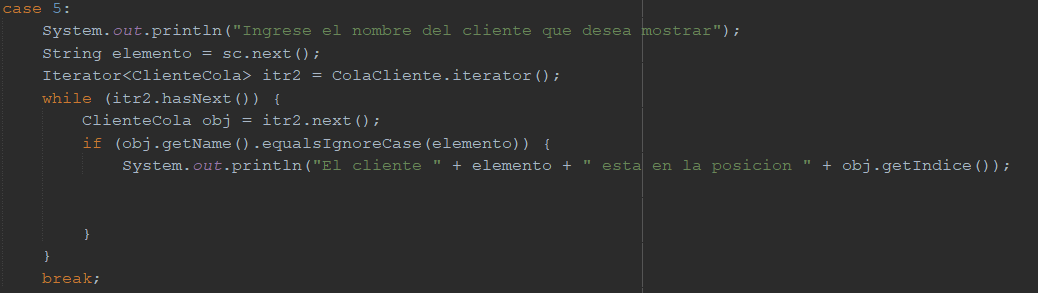
1. Explique textual y gráficamente cual es el algoritmo o proceso para **sacar el próximo elemento** en una Cola. Hacer un ejemplo en Java.

Texto

Descripción generada automáticamente

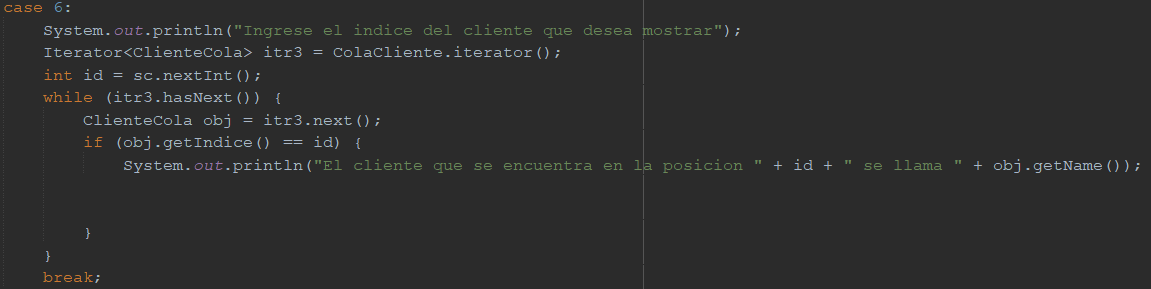
Queue y LinkedList proveen el método poll() el cual permite sacar el primer cliente de la cola.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **obtener la posición de un elemento** en una Cola. Hacer un ejemplo en Java.



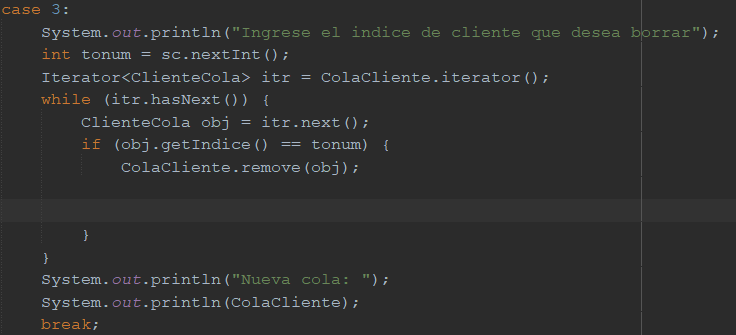
Al igual que en el punto 9, pedimos el nombre que desean revelar su posición. Recorremos la cola mediante un Iterator y le decimos al iterador que cuando encuentre un nombre igual que regrese el índice almacenado.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **obtener un elemento en la posición X de** una Cola. Hacer un ejemplo en Java.



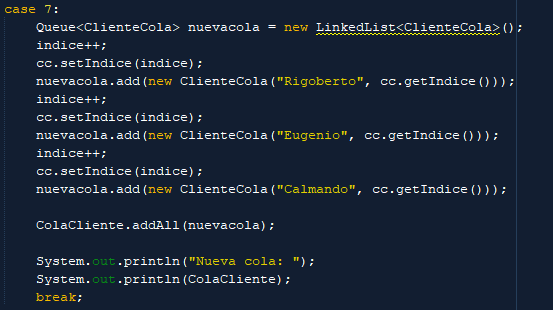
En este punto pedimos por consola el índice que desean buscar, a su vez creamos un Iterator que recorrerá la cola y cuando este encuentre un índice igual en la cola, que haga un print en donde regrese el nombre del cliente con dicho índice.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **eliminar** **un elemento cualquiera** de una Cola. Hacer un ejemplo en Java.



Queue y LinkedList poseen un método llamado .remove() el cual elimina dicho elemento de la cola, por tanto, pedimos el índice del cliente que desean eliminar de la cola. Recorremos la cola mediante un Iterator

1. Explique textual y gráficamente cual es el algoritmo o proceso para **agregar** **varios** elementos a una **Cola,** es decir, **agregar una subCola en otra Cola.** Hacer un ejemplo en Java.



Queue y LinkedList poseen un método llamado .addAll() el cual añade una cola entera al final de otra cola, por tanto, creamos una nueva cola de tipo ClienteCola y añadimos datos usando el método .add() usado en el punto 10.

Luego le decimos a la cola ColaCliente que adquiera todos los elementos de la nueva cola usando addAll().

Por último, hacemos print de la ColaCliente con sus nuevos elementos.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **eliminar** **varios** elementos consecutivos o secuenciales de **cualquier parte de la Cola, e**s decir, **eliminar una subCola en otra Cola, pero que no sea del final de la cola.** Hacer un ejemplo en Java.

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

En este punto pedimos dos índices, el índice inicial y el índice final de los elementos que queremos borrar. Queue y LinkedList nos ofrece un método llamado removeIf() el cual necesita de una condición para funcionar.

Para hacer dicha condición tenemos que importar el paquete predicate en donde implementamos un predicator llamado n que eliminará los elementos de la cola desde el índice inicial hasta el índice final.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **borrar** **todos los elementos** de una Cola, es decir, vaciar una Cola. Hacer un ejemplo en Java.

Logotipo

Descripción generada automáticamente

Para borrar todos los elementos de una cola, Queue y LinkedList nos ofrecen el método clear() el cual elimina todos los elementos de la cola.

**PILAS**:

1. Explique textual y gráficamente que es un una Pila, y cuáles son las clases ofrece el SDK de Java para trabajar con Pilas

Las pilas (también llamadas stack) son un conjunto de elementos ordenados donde solo se puede ingresar y eliminar elementos desde un solo extremo llamado cima (o top) que se encuentra en dicha pila. Debido a esto, las pilas son consideradas estructuras de datos LIFO (Last in, First Out).

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Tomado de: https://es.wikipedia.org/wiki/Pila\_(informática)

Las pilas poseen ciertas operaciones básicas como:

* Apilar()
* Desapilar()
* CrearPila()
* Cima()

Ciertamente java ofrece una clase llamada Stack, la cual facilita el trabajar con pilas en Java. Sus operación básicas son push(introduce un elemento), pop(saca un elemento), peek(regresa el elemento que esta en la cima), empty(comprueba si la pila esta vacia) y search(la cual busca un elemento en la pila y regresa su posicion).

1. Explique por qué las pilas se consideran elementos fundamentales en las estructuras de datos.

Al igual que las colas, las pilas son de vital importancia dentro de la programación debido a que propician distintas formas de organización y manejo de datos que poseen cierta relación o cierto valor.

De tal manera que permiten el procesamiento eficiente de grandes cantidades de información.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **contar los elementos de una Pila, desde el primero hasta el último.** Hacer un ejemplo en Java

Texto

Descripción generada automáticamente con confianza baja

Debido a que implementamos la pila usando ArrayDeque, esta clase posee métodos para determinar el numero de elementos. En este caso hacemos un print del tamaño de la pila, usando el método .size().

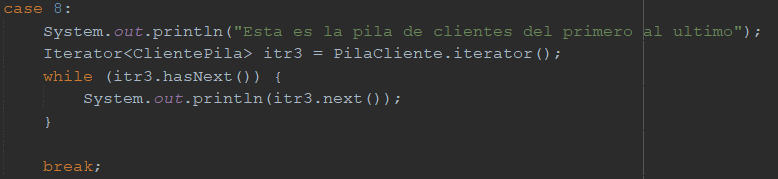
1. Explique textual y gráficamente cual es el algoritmo o proceso para **saber si una Pila está vacía o no.** Hacer un ejemplo en Java

Texto

Descripción generada automáticamente

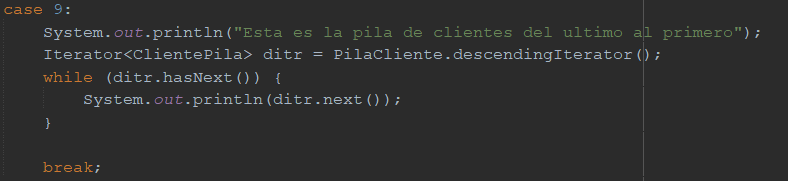
ArrayDeque nos ofrece un método para saber si la pila esta vacia o no, por tanto, usamos un if para evaluar si el método .isEmpty() regresa true o false, si este regresa true significa que esta vacia y si regresa false es porque tiene elementos.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **recordar o mostrar los elementos de una Pila, desde el primero hasta el último.** Hacer un ejemplo en Java



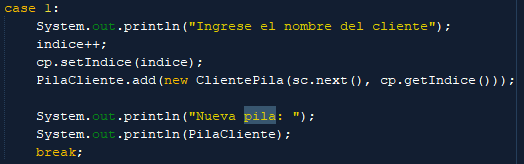
La clase ArrayDeque (al igual que Queue y LinkedList) nos ofrece un método llamado .iterator() para iterar nuestra pila. Por tanto en este punto creamos un iterador que recorra la pila y que a su vez la imprima en consola.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **recorrer o mostrar los elementos de una Pila, desde el último hasta el primero.** Hacer un ejemplo en Java



La clase ArrayDeque (al igual que Queue y LinkedList) nos ofrece un método llamado .descendingIterator para iterar nuestra pila en reversa. Por tanto, en este punto creamos un iterador invertido que recorra la pila y que a su vez la imprima en consola.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **agregar un elemento a una Pila.** Hacer un ejemplo en Java



ArrayDeque provee el método add() el cual añade un objeto a la pila. La pila es de tipo ClienteCola (String, int) por tanto para añadir un cliente pedimos el nombre por teclado y su índice es el incremento del indice guardado en el setIndice().

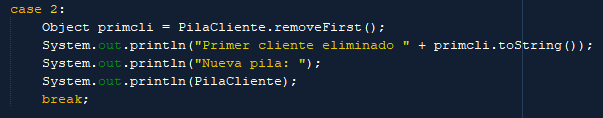
1. Explique textual y gráficamente cual es el algoritmo o proceso para **insertar un elemento a una Pila.** Hacer un ejemplo en Java

Texto

Descripción generada automáticamente

ArrayDeque provee el método add() el cual añade un objeto a la pila. La pila es de tipo ClientePila (String, int) por tanto para añadir un cliente pedimos el nombre por teclado y su índice es el incremento del indice guardado en el setIndice().

1. Explique textual y gráficamente cual es el algoritmo o proceso para **eliminar el primer elemento de una pila.** Hacer un ejemplo en Java



ArrayDeque nos ofrece el método removeFirst() el cual nos permite eliminar el primer elemento de la pila. Sin embargo, para poder mostrar dicho elemento debimos crear un objeto llamado primcli para imprimir que elemento se había eliminado.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **eliminar el último elemento de una pila.** Hacer un ejemplo en Java

Texto

Descripción generada automáticamente

ArrayDeque nos ofrece el método removeLast() el cual nos permite eliminar el ultimo elemento de la pila. Sin embargo, para poder mostrar dicho elemento debimos crear un objeto llamado ulticli para imprimir que elemento se había eliminado.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **buscar un elemento de una pila.** Hacer un ejemplo en Java\*\*\*

Texto

Descripción generada automáticamente

En este caso, pedimos por consola que coloquen el nombre del cliente que desean verificar en la pila. Recorremos la cola mediante un Iterator (que hace parte de los métodos presentes en ArrayDeque) y le decimos al iterador que cuando encuentre un nombre igual al ingresado, diga que ese cliente si está en la pila.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **agregar varios elementos consecutivos o secuenciales en cualquier posición de la Pila.** Hacer un ejemplo en Java

Texto

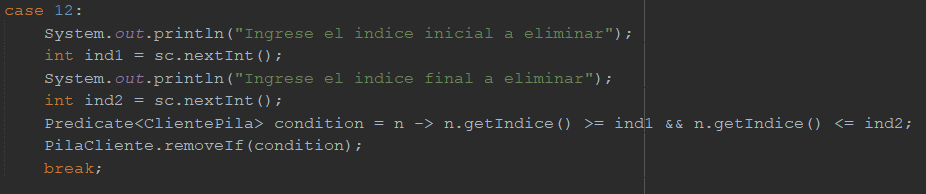
Descripción generada automáticamente

ArrayDeque posee un método llamado .addAll() el cual añade una pila entera al final de otra pila, por tanto, creamos una nueva pila de tipo ClientePila y añadimos datos usando el método .add() usado en el punto 7 y 8.

Luego le decimos a la pila PilaCliente que adquiera todos los elementos de la nueva pila usando addAll().

Por último, hacemos print de la PilaCliente con sus nuevos elementos.

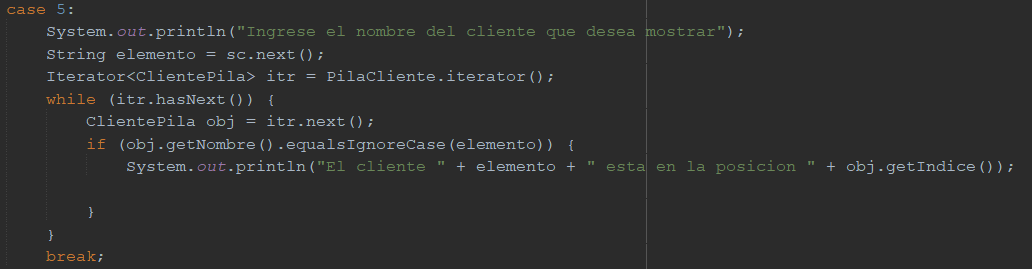
1. Explique textual y gráficamente cual es el algoritmo o proceso para **eliminar varios elementos consecutivos o secuenciales en cualquier posición de la Pila.** Hacer un ejemplo en Java



En este punto pedimos dos índices, el índice inicial y el índice final de los elementos que queremos borrar. ArrayDeque nos ofrece un método llamado removeIf() el cual necesita de una condición para funcionar.

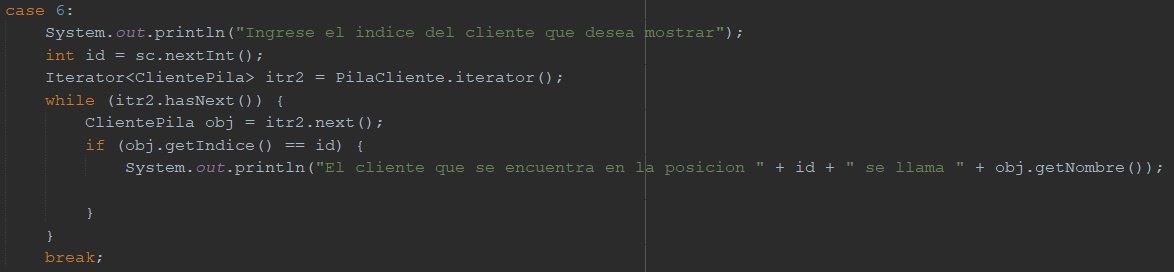
Para hacer dicha condición tenemos que importar el paquete predicate en donde implementamos un predicator llamado n que eliminará los elementos de la pila desde el índice inicial hasta el índice final.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **obtener la posición de un elemento de la Pila.** Hacer un ejemplo en Java



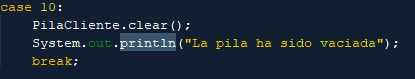
Al igual que en el punto 11, pedimos el nombre que desean revelar su posición. Recorremos la pila mediante un Iterator y le decimos al iterador que cuando encuentre un nombre igual que regrese el índice almacenado.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **obtener un elemento a partir de una posición específica de la Pila.** Hacer un ejemplo en Java



En este punto pedimos por consola el índice que desean buscar, a su vez creamos un Iterator que recorrerá la pila y cuando este encuentre un índice igual almacenado en la pila, que haga un print en donde regrese el nombre del cliente con dicho índice.

1. Explique textual y gráficamente cual es el algoritmo o proceso para **Eliminar todos los elementos (vaciar) de una Pila.** Hacer un ejemplo en Java



ArrayDeque nos ofrece un método llamado clear(), el cual permite eliminar todos los elementos presentes en la pila.

Bibliografía

<http://agrega.juntadeandalucia.es/repositorio/02122016/a5/es-an_2016120212_9131705/34_colas.html>

<http://aprenderestructuradedatos.blogspot.com/2015/11/colas-en-java.html>

<https://www.uaeh.edu.mx/docencia/P_Presentaciones/icbi/asignatura/Cap3PilasColas.pdf>

<http://agrega.juntadeandalucia.es/repositorio/02122016/a5/es-an_2016120212_9131705/32_pilas.html>

<https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=608:la-estructura-de-datos-pila-en-java-clase-stack-del-api-java-ejemplo-simple-y-ejercicios-resueltos-cu00923c&catid=58&Itemid=180>

<https://spa.myservername.com/java-priority-queue-tutorial-implementation-examples>

<http://pilascolasenjavaestructuradedatos.blogspot.com/2016/05/colas-con-prioridad-en-java.html>

<https://www.softwaretestinghelp.com/java-queue-interface/>

<https://w3api.com/Java/ArrayList/removeIf/>

<https://docs.oracle.com/javase/7/docs/api/java/util/ArrayDeque.html>

<https://docs.oracle.com/javase/8/docs/api/java/util/Iterator.html>