

**UNIVERSIDAD TECNOLÓGICA CENTROAMERICANA**

**UNITEC**

**ASIGNATURA: ESTRUCTURAS DE DATOS I**

**SECCIÓN­: 489**

**PROYECTO INDIVIDUAL I**

**TDAs FUNDAMENTALES**

**PRESENTADO POR:**

**ALUMNO: JUDÁ AARON PONCE VILLALTA**

**CTA: 11841248**

**CATEDRÁTICO:** ING**. REYNOD BOCANEGRA**

**CAMPUS TEGUCIGALPA 31 DE AGOSTO DE 2020**

**Introducción**

El siguiente manual mostrará cada uno de los metodos utilizados en la elaboración del proyecto. Dicho proyecto está hecho en c++ con IDE netbeans, que muestra la implementación de los tipos de datos abstractos con arreglos y nodos doblemente enlazados.

Se mostrará cada detalle usando capturas de pantalla y pseudocodigos de cada metodo para tener un mayor beneficio en cuanto al entendimiento de lo que se elaboró a lo largo del proyecto.

Jerarquía de clases

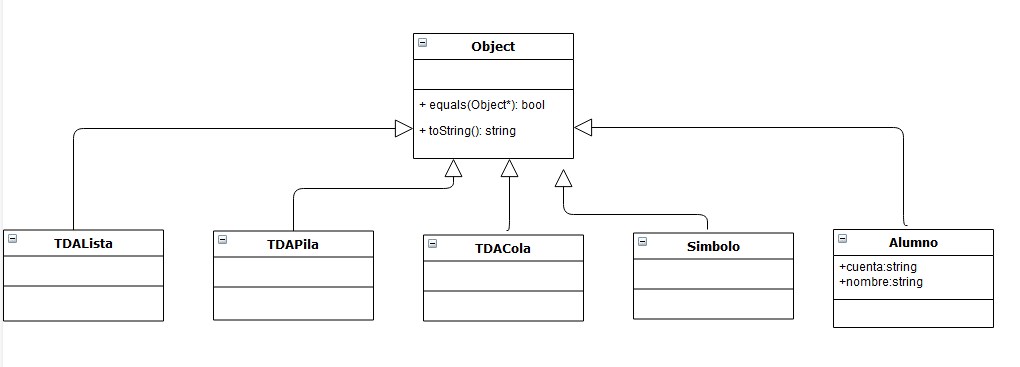


Imagen que muestra la composición de cada una de las clases que serán utilizadas a lo largo del proyecto. Cabe mencionar que hay existencia de 6 clases más que serán heredadas de las deferentes TDAs, en la cuales tenemos:

1. Para TDA Lista:

* Arraylist (con arreglos)
* Linked List (con nodos)

1. Para TDA Pila

* Array stack (con arreglos)
* Linked stack (con nodos)

1. Para TDA Cola

* Array queue (con arreglos)
* Linked queue (con nodos)

Main del proyecto

**Una captura de pantalla de un celular con letras

Descripción generada automáticamenteCaptura de pantalla de un celular con letras

Descripción generada automáticamente**

Inicialmente se deben de incluir todas las clases antes mencionadas en el main, así como también la creación de los prototipos de las funciones a utilizar, de forma opcional se puede hacer el uso de variables globales.

En el main, los prototipos de las funciones lo que contiene cada una de ellas es el manejo de los diferentes menús a utilizarse como se muestran en las imágenes, a conveniencia propia, las funciones de los menús pueden ser llamados dentro de otras funciones que tengan que ver con algún otro menú, por ejemplo, el menú para escoger las opciones de arraylist perfectamente puede ser llamado en el menú de listas y así sucede con los otros.

Hacer las instancias de los nuevos objetos que se utilizaran mediante apuntadores y finalmente cuando ya se haya hecho el trabajo deseado, hacer la respectiva limpieza de memoria.

Implementación de la clase “Object”

Captura de pantalla de un celular

Descripción generada automáticamente

Como se muestra en la imagen, la clase Object es una clase abstracta, quiere decir que no se puede instanciar, esta clase contiene 2 funciones polimórficas que se explicaran mas adelante. Es de vital importancia que esta clase sea la primera en crearse, ya que todas las clases tienen un cierto de dependencia de esta. Con esta clase ya no hay mas nada que hacer, y se prosigue a la creación de las siguientes clases de las diferentes TDAs.

A continuación, se presentan la implementación de las clases complementarias:

Implementación de la clase “Nodo”

Captura de pantalla de un celular

Descripción generada automáticamente Imagen que contiene captura de pantalla

Descripción generada automáticamente

Esta clase será utilizada para implementar nodos doblemente enlazados donde se requiera, solo se debe incluir la clase object, ya que los datos que almacenará cada nodo es de tipo objeto.

Implementación de la clase “Alumno”

Una captura de pantalla de una red social

Descripción generada automáticamente Una captura de pantalla de una computadora

Descripción generada automáticamente

Esta clase servirá como objeto para trabajar con TDA Lista y TDA Cola respectivamente, alumno hereda de Object, así como también sus método polimórficos “toString” e “equals”, como se muestran en las imágenes.

Implementación de la clase “Símbolo”

Captura de pantalla de un celular

Descripción generada automáticamente Una captura de pantalla de una red social

Descripción generada automáticamente

Esta clase al igual que alumno hereda objeto, que se utilizara para TDA Pila, también hereda sus 2 métodos polimórficos de la clase objeto.

Implementación de TDA Lista

Captura de pantalla de un celular

Descripción generada automáticamente Captura de pantalla de un celular

Descripción generada automáticamente

Como se muestran en las imágenes, la clase TDA LIST, no hereda de object pero es necesario incluirla ya que como en algunas funciones retornan datos de tipo object o recibe paramatros de tipo Object. TDA Lista es una clae abstracta y la vez es clase padre de las clases arraylist y linkedlist.

Implementación de la clase arraylist

Captura de pantalla de un celular

Descripción generada automáticamente

Como mencionaba anteriormente, la clase arraylist hereda de TDA Lista, así como todos los métodos, esta clase será implementada con arreglos. Como se observa en la imagen esta clase tiene tres atributos con encapsulamiento de tipo protected, tenemos un atributo un arreglo de apuntadores tipo Object y dos atributos de tipo entero, capacity, que controlará la capacidad del arreglo y size que es el tamaño de elementos contenidos en el mismo.

A continuación, se explicarán más a detalle cada uno de los métodos y como funcionan cada uno de ellos.

Método insertar (Arraylist)

Una captura de pantalla de una red social

Descripción generada automáticamente

Esta función de tipo booleana que recibe de parámetro un entero que es la posición a insertar y el objecto (en este caso de tipo alumno), primero verifica que la posición seleccionada sea valida en el rango establecido al momento, si se cumple esa condición, entra al primer if, luego hay una serie de ifs anidados que dependiendo el valor de p hará su función, si p es igual a 1, quiere decir que el objecto será insertado al inicio, se hace el corrimiento con un for al revés y guarda en la posición 0 el objeto, si el valor p es igual que la cantidad de elementos, quiere decir que este será insertado al final del arraylist, y este no necesita hacer ningún corrimiento, finalmente la ultima condición que implícitamente se entiende que si no fue al inicio ni al final entonces se inserta en medio, se hace un corrimiento con un un for al revés, ya que así es menos probable que se pierdan algunos valores, hacer el incremento de size y retornar verdadero, si no se retorna falso.

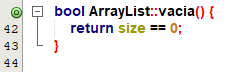
Método suprimir (Arraylist)

Captura de pantalla con letras y números

Descripción generada automáticamente

Esta función retorna Object y recibe de parámetro un entero que es la posición, lleva casi la misma lógica que insertar solo que a la inversa, primero se valida que el valor ingresado este dentro del rango permitido, si es así, entonces se debe crear una variable temporal de tipo object para capturar el valor de p-1, si el usuario decide borrar el ultimo elemento entonces entrará al primer if, en este caso no debe de hacer ningún corrimiento, solo decrementar size, si no es así se hace un corrimiento desde el valor de p-1 hasta la size, y al arreglo en la posición i se le asigna lo que esté en la posición i+1, y a la posición size-1 se le asigna NULL y se decrementa el size y se retorna la variable temporal si no se retorna NULL.

Método vacía (Arraylist)

Método booleano, no recibe nada de parámetro, retorna verdadero si el arreglo tiene 0 elementos, si no retorna falso.

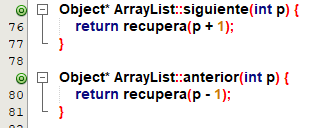
Método recupera (Arraylist)

Una captura de pantalla de un celular con letras

Descripción generada automáticamente

Función que retorna un object y recibe de parámetro un entero que es la posición ingresada por el usuario, primero se debe de crear una variable temporal de tipo object y asignarle NULL, validar que p este en el rango permitido, de ser así, a la variable retval se le asigna lo que este en el array p-1 y lo retorna, si no retorna retval con el valor de NULL.

Métodos siguiente y anterior (Arraylist)

Funciones que retornan object y reciben un entero que es la posición que el usuario ingresó, se hace el llamado del método recupera y para el método el siguiente retorna el método recupera con la posición p+1, y para el método anterior retorna el método recupera con la posición p-1.

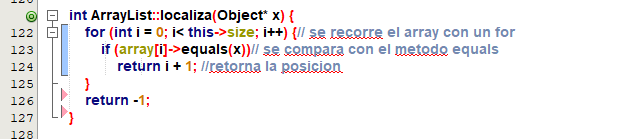
Método imprimir (Arraylist)

Captura de pantalla con letras y números

Descripción generada automáticamente

Función que no retorna nada, primero valida con el método vacía, que no este vacío el, si esta vacío muestra un mensaje, si no con un for que recorre todo lo que este en el arreglo y se hace el llamado del método toString, que será explicado mas adelante e imprime los datos.

Método localiza (Arraylist)



Función que retorna un entero y recibe de parámetro un objecto, mediante un for iterará y buscará x mediante una validación y el uso del método equals, que se explicará mas adelante, si es igual a x entonces retorna la posición en la que se encuentra ese objeto, si no retornará el valor de -1.

Implementación de la clase Linked List

Captura de pantalla de un celular

Descripción generada automáticamente

Como mencionaba anteriormente, la clase linked list hereda de TDA Lista, así como todos los métodos, y su vez se debe de incluir la clase nodo ya que esta clase será implementada con el uso de nodos doblemente enlazados. Como se observa en la imagen esta clase tiene tres atributos con encapsulamiento de tipo private, dos atributos de tipo nodo que indican el inicio y el final de la lista, y un atributo de tipo entero, n que es la cantidad de nodos en la lista.

A continuación, se explicarán más a detalle cada uno de los metodos y cómo funcionan cada uno de ellos.

Método insertar (Linked list)

Captura de pantalla de un celular

Descripción generada automáticamenteFunción booleana que recibe un entero que es la posición ingresada por el usuario, y el objeto a insertar, se valida que p este en el rango, de ser así, se crea una variable temporal de tipo nodo, y esa variable se le asigna x, luego valida si el inicio del nodo es igual a NULL entonces solo ahí se puede insertar el objeto, luego al siguiente nodo del inicio se le asigna NULL así como también al nodo anterior, luego a fin se le asigna inicio ya que en ese momento solo tendrá un nodo, y se incrementa n, si no al fin de la lista se le asigna la variable temporal anteriormente creada, y a esa misma variable sus nodos que apuntan a siguiente y anterior respectivamente se les asigna NULL, luego al final se le asigna la “nueva”, se incrementa n y se retorna verdadero, si no retorna falso.

Método suprimir (Linked list)

Captura de pantalla de un celular con letras

Descripción generada automáticamente Captura de pantalla de un celular con letras

Descripción generada automáticamente

Método vacía (Linked list)

Captura de pantalla de un celular con letras

Descripción generada automáticamenteFunción booleana que retorna verdadero si n es igual a 0, si no retorna falso.

Método recupera (Linked list)

Captura de pantalla de un celular

Descripción generada automáticamenteFunción que retorna un objeto y recibe de parámetro un entero que hace referencia a la posición ingresada por el usuario, inicialmente se debe de crear una variable entera que son los saltos que dará el nodo asignándole p-1, se valida que p sea válido, si es válido, se crea una variable temporal de tipo nodo y se le asigna el inicio de la lista, luego un for de 1 hasta hops, y a la variable temporal se le asigna el nodo siguiente, y se retorna esa variable con sus datos respectivos, si no se retorna NULL.

Métodos siguiente y anterior (Linked list)

Captura de pantalla de un celular

Descripción generada automáticamente Funciones que retornan object y reciben un entero que es la posición que el usuario ingresó, se hace el llamado del método recupera y para el método el siguiente retorna el método recupera con la posición p+1, y para el método anterior retorna el método recupera con la posición p-1.

Método imprimir (Linked list)

Una captura de pantalla de un celular con letras

Descripción generada automáticamente

Función que no retorna nada, e inicialmente se hace el llamado del método vacia que sirve como validación, si el método vacia retorna falso, entonces se crea una variable temporal de tipo nodo a la cual se le asigna el inicio de la lista, y en un ciclo while, mientras la variable actual sea diferente de NULL entonces se imprime el objeto que esta contenido en la lista usando el método toString y el método getData, luego a esa variable se le asigna siguiente hasta llegar a NULL o hasta llegar al final de la lista.

Método localiza (Linked list)

Una captura de pantalla de un celular con letras

Descripción generada automáticamenteFunción que retorna un entero, que recibe de parámetro un objeto, se crea una variable temporal de tipo entero, y se le asigna n-1, así como también una variable temporal de tipo nodo a la cual se le asigna el inicio de la lista.

Iterar con un for desde 1 hasta hops, y se valida que el objeto ingresado sea igual al objeto que este contenido en el nodo y retorna la posición donde fue encontrado, luego a la variable temporal se le asigna siguiente para verificar si en el siguiente nodo está el objeto con los mismo datos, de no ser así, retornará el valor de -1.

Implementación de la clase “TDA Pila”

Captura de pantalla de un celular

Descripción generada automáticamenteCaptura de pantalla de un celular

Descripción generada automáticamente

Como se muestran en las imágenes, la clase TDA Pila, no hereda de object pero es necesario incluirla ya que como en algunas funciones retornan datos de tipo object o recibe paramatros de tipo object. TDA Pila es una clase abstracta y la vez es clase padre de las clases ArrayStack y LinkedStack.

Implementación de la clase Arraystack

Captura de pantalla de un celular

Descripción generada automáticamente

Como mencionaba anteriormente, la clase arraystack hereda de TDA Pila, así como todos los métodos, esta clase será implementada con arreglos. Como se observa en la imagen esta clase tiene tres atributos con encapsulamiento de tipo protected, tenemos un atributo un arreglo de apuntadores tipo Object y dos atributos de tipo entero, capacity, que controlará la capacidad del arreglo y tope que es el tamaño de elementos contenidos en el mismo.

A continuación, se explicarán más a detalle cada uno de los métodos y como funcionan cada uno de ellos.

Método empujar (push)

Imagen que contiene cuchillo

Descripción generada automáticamenteFunción de tipo void, que recibe como parámetro un objeto, en esta caso el objeto será de tipo símbolo, luego ya que en el constructor tanto el arrayPila = NULL y tope=0, entonces lo que esta en valor de tope se le asigna x, y luego se incrementa el tope.

Método isEmpty

Imagen que contiene cuchillo

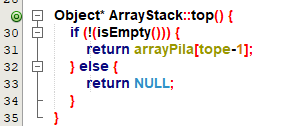
Descripción generada automáticamenteFunción booleana que retornará verdadero si la pila tiene 0 elementos o falso si al menos hay 1 elemento o mas dentro de la pila.

Método sacar (pop)

Captura de pantalla de un celular con letras

Descripción generada automáticamenteFunción que retornara un objeto, y no recibe nada de parámetro, se valida primero que la pila no está vacía con el llamado del método isEmpty(), si lo está, retornara el valor de NULL, si no lo está se crea una variable temporal de tipo object y se captura el valor del tope-1 e indemadiatamente después al arreglo en la casilla con el valor de tope-1 se le asigna NULL, se decrementa tope en 1 y se retorna la variable temporal.

Método ver tope (top)

Función que retornará un objeto y no reciba nada de parámetro, se valida primero que la pila no esté vacía para así retornar lo que este en la casilla tope-1, ya que se mostrará el último elemento insertado.

Método imprimir (arraystack)

Imagen que contiene pájaro

Descripción generada automáticamente

Funcion que no retorna nada, se valida que la pila no este vacia, si lo esta se muestra un mensaje, si no con un for desde tope-1 hasta 0, se recorre el for así porque se quiere imprimir desde el ultimo hasta el primero, se imprime lo que esta contenido en el arreglo y a la vez se hace el llamado del método toString() y listo.

Implementación de la clase Linked stack

Captura de pantalla de un celular

Descripción generada automáticamente

Como mencionaba anteriormente, la clase linked stack hereda de TDA Pila, así como todos los métodos, y su vez se debe de incluir la clase nodo ya que esta clase será implementada con el uso de nodos doblemente enlazados. Como se observa en la imagen esta clase solo tiene 1 atributo con encapsulamiento de tipo protected, el atributo de tipo nodo que indica el tope de la lista.

A continuación, se explicarán más a detalle cada uno de los métodos y cómo funcionan cada uno de ellos.

Método empujar (push) Linked stack

Captura de pantalla con letras y números

Descripción generada automáticamente Funcion que no retorna nada y recibe como parámetro un objeto, primero se debe de crear una nueva variable de tipo nodo, y a ese nuevo nodo asignarle el objeto que ha sido ingresado, y el nodo siguiente se le asigna tope que inicialmente está en NULL en el constructor, y al nodo anterior se le asigna nuevo nodo, y finalmente a tope se le asigna el nuevo nodo.

Método isEmpty (Linked stack)

Imagen que contiene cuchillo

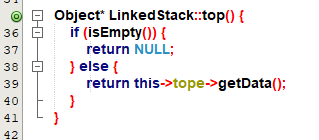
Descripción generada automáticamente Función booleana que retornará verdadero si el tope de la pila es igual a NULL o falso si al menos hay 1 elemento o más dentro de la pila.

Método sacar (pop) Linked list

Una captura de pantalla de un celular con letras

Descripción generada automáticamenteFunción que retorna un objeto, primero se valida si la pila esta vacia, de ser así, retorna NULL, si no se crean 2 variables temporales, una de tipo nodo que se le asigna tope, y la otra de tipo Object que se le asigna NULL, luego a tope recibe lo que este en el nodo siguiente, luego el nodo anterior recibe el valor de NULL, así como también el nodo siguiente, esto para que se este apuntando correctamente, luego a retval recibe el dato, y temp se le asigna NULL, y finalmente retorna retval y finalmente se saca el dato de la pila.

Método ver tope (top) en Linked stack

Funcion que retorna un objeto, se valida primero que la pila no este vacia, si lo esta se retorna NULL, si no lo está se retorna el dato que está en el nodo tope, ya que se mostrará el último símbolo ingresado por el usuario.

Método imprimir (Linked stack)

Una captura de pantalla de un celular con letras

Descripción generada automáticamenteFunción que no retorna nada, primero se verifica que no este vacia, si lo esta se muestra un mensaje que la pila esta vacia, si no se crea una variable temporal de tipo nodo y se le asigna el tope de la pila, luego mientras ese nodo no sea NULL, dentro de un ciclo while se mostrara el dato que contiene ese nodo, y luego a la variable temporal recibe el nodo siguiente y así hasta encontrar un nodo= NULL.

Implementación de la clase “TDA Cola”

Captura de pantalla de un celular

Descripción generada automáticamenteCaptura de pantalla de un celular

Descripción generada automáticamente

Como se muestran en las imágenes, la clase TDA Cola, no hereda de object pero es necesario incluirla ya que como en algunas funciones retornan datos de tipo object o recibe paramatros de tipo object. TDA Cola es una clase abstracta y la vez es clase padre de las clases ArrayQueue y LinkedQueue.

Implementación de la clase ArrayQueue

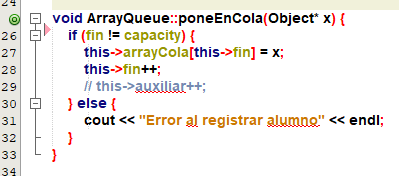
Captura de pantalla de un celular

Descripción generada automáticamente

Como mencionaba anteriormente, la clase arrayqueue hereda de TDA Cola, así como todos los métodos, esta clase será implementada con arreglos. Como se observa en la imagen esta clase tiene cinco atributos con encapsulamiento de tipo protected, tenemos un atributo un arreglo de apuntadores tipo Object y cuatro atributos de tipo entero, capacity, que controlará la capacidad del arreglo, front que será para controlar el frente de la cola, fin para controlar el final de la cola, y auxiliar por si en algún momento debemos guardar valores que no queremos que se pierdan en tiempo de ejecución, estas variables front y fin sirven también para ver cual es el tamaño de elementos contenidos en el mismo.

A continuación, se explicarán más a detalle cada uno de los métodos y como funcionan cada uno de ellos.

Método “Encolar” (queue) ArrayQueue

Funcion que no retorna nada y recibe como parámetro un objeto, se valida primero que la cola no esté llena, si se cumple esa condición, al arreglo en la casilla con el valor de fin se le asigna x y se guarda, se aumenta fin para poder seguir guardando datos en la casilla que sigue y así sucesivamente, si no se cumple la condición, se muestra un mensaje que no se ha podido registrar.

Método vacia ArrayQueue

Imagen que contiene cuchillo

Descripción generada automáticamenteFuncion booleana que retorna verdadero si front y fin tienen el mismo valor, ya que si lo tiene entonces no hay elementos en la cola, si no retorna falso.

Método desencolar (dequeue) ArrayQueue

Captura de pantalla de un celular

Descripción generada automáticamenteFuncion que retorna object, inicialmente se debe validar que la cola no este vacia, si no lo está, se crea una variable temporal de tipo Object que se le asigna lo que esta en el arreglo en la casilla front, porque la cola se comporta de esa manera, el primero va saliendo, se incrementa front y se retorna retval, si está vacia se retorna NULL.

Método ver tope (peek) ArrayQueue

Captura de pantalla con letras y números

Descripción generada automáticamenteFuncion que retorna un tipo de dato object que no recibe nada de parámetro, se verifica que la cola no este vacia, y si no lo está, con una variable temporal de tipo object se le asigna lo que este en el arreglo en la casi front, y lo retorna, si esta vacia, retorna NULL.

Método imprimir ArrayQueue

Una captura de pantalla de un celular con letras

Descripción generada automáticamente

Funcion de tipo void, primero se valida que la cola no este vacia, si no lo está, con un ciclo for se itera desde el frente hasta el fin de la cola, se imprime lo que se encuentre y muestra los datos con el método toString, si esta vacia se muestra un mensaje que la cola esta vacia.

Implementación de la clase LinkedQueue

Captura de pantalla de un celular

Descripción generada automáticamente

Como mencionaba anteriormente, la clase linked queue hereda de TDA Cola, así como todos los métodos, y su vez se debe de incluir la clase nodo ya que esta clase será implementada con el uso de nodos doblemente enlazados. Como se observa en la imagen esta clase solo tiene 2 atributos con encapsulamiento de tipo protected, el atributo de tipo nodo que indica el inicio y el fin de la cola.

A continuación, se explicarán más a detalle cada uno de los métodos y cómo funcionan cada uno de ellos.

Método “Encolar” (queue) LinkedQueue

Captura de pantalla de un celular con letras

Descripción generada automáticamenteFuncion de tipo void que recibe como parámetro un objeto, primero se crea una variable nueva de tipo nodo a la cual a esta misma se le asigna x, o sea el objeto, en la validación es controlada por el método vacia, significa que si esta vacia, el inicio y el fin de la cola tendrán el mismo dato, de no ser así, a la variable fin apuntara al nodo siguiente que recibe el nuevo nodo y la variable nueva de tipo nodo, apuntara al nodo anterior que recibe el fin del nodo, luego a fin se le asigna nuevo.

Método vacia LinkedQueue

Imagen que contiene cuchillo

Descripción generada automáticamenteFuncion booleana que retornará verdadero si el nodo inicial es NULL, quiere decir que no hay ningún elemento en la cola, si hay 1 o mas elementos retornará falso.

Método desencolar (dequeue) LinkedQueue

Captura de pantalla de un celular

Descripción generada automáticamenteFuncion de tipo Object, primero se deben crear 2 variables auxiliares o temporales de tipo nodo, a la variable llamada actual se le asigna el inicio y la variable llamada anterior se le asigna NULL. Mediante una validación si el inicio es distinto de NULL entonces el inicio recibe lo que esta en el nodo siguiente, si la variable actual es igual a fin, entonces el nodo anterior apuntará al nodo anterior que recibe como valor NULL, luego a fin se le asigna lo que esté en anterior. Si no, anterior apuntará al nodo siguiente que recibe lo que este apuntando la variable actual(siguiente) luego la variable anterior se le asigna lo que este en actual para no perder el dato, y actual se le asigna lo que este en el nodo siguiente, si no retorna NULL.

Método ver tope (peek) LinkedQueue

Captura de pantalla con letras y números

Descripción generada automáticamenteFuncion de tipo Object, no recibe nada de parámetro, primero se verifica que la cola no está vacía, si lo está, retorna NULL, si no lo está, se retorna lo que este en el nodo inicial con su dato usando el método getData().

Método imprimir LinkedQueue

Imagen que contiene pájaro, ave

Descripción generada automáticamente

Función de tipo void, primero debe crear una variable actual que se encargará de hacer una buena iteración, en la cual se le asigna el inicio de la lista, se valida que no esté vacía, si no lo está entonces mientras la variable actual sea distinta de NULL, se imprime el dato que esté en el nodo y muestra lo que tiene el dato con el método toString(), luego el nodo actual pasará al nodo siguiente para poder imprimir mas elementos si lo hay, si la cola está vacía se muestra un mensaje de que está vacía.