ESCUELA POLITÉCNICA NACIONAL

Carrera Software

Nombre: Fernando Eliceo Huilca Villagómez Fecha:18/05/2024 Grupo:GR2

Profesora: Dra. Mayra CARRION

ESTRUCTURAS DE DATOS TIPO COLA CIRCULAR

1. **PROBLEMA 1:** TRABAJAR LA IMPLEMENTACION DE LOS ALGORITMOS DE OPERACIONES BASICAS DE LA ESTRUCTURA DE DATOS TIPO COLA CIRCULAR

   a. **ALGORITMOS OPERACIONES BÁSICAS**

```
incio AlgoritmoEliminarDato
  si(colaEstáVacía())
        Escribir "La cola está vacía.
        Desbor"
  Sino
        colaCircular[frente]=null
        si(frente = fin)
        frente = -1 y fin = -1
        sino si (frente = max)
        frente = 0
                sino
                frente++
  Fin AlgoritmoEliminarDato
```

```
        incio AlgoritmoInsertarDato
si(colaEstáLlena())
        Escribir "La cola está llena.
        Desbor"
Sino Si (fin = max)
          fin = 0
      sino fin++
colaCircular[fin] <- Dato
    Fin AlgoritmoInsertarDato
```

InicioAlgoritmoSiColaEstaVacía

Si frente == -1

Return verdad

Sino

Retur falso

FinAlgoritmoSiColaEstaVacía

InicioAlgoritmoSiColaEstaLlena

Si (frente = 0 y fin = max) o (fin +1 == frente)

Return verdad

Sino

Retur falso

FinAlgoritmoSiColaEstaLlena

b. IMPLEMENTACION

En la siguiente imagen se muestra la creacion de una clase llamada “ColaCircular”. La clase ColaCircular tiene cuatro atributos privados: frente, fin, maximo y un array de Strings llamado colaCircular. El constructor recibe un parámetro entero tamanioCola para establecer el tamaño del array colaCircular e inicializa maximo a tamanioCola - 1, y tanto frente como fin a -1. El comentario sobre la declaración del array indica que este ejemplo utilizará una cola de tipo String.

```
public class ColaCircular {
    // Atributos de la clase:
    private int frente;
    private int fin;
    private int maximo;
    private String[] colaCircular; // En este ejemplo vamos a usar una cola de tipo St

    public ColaCircular(int tamanioCola) { // Constructor de la clase colaCircular que
        colaCircular = new String[tamanioCola];
        maximo = tamanioCola - 1;
        frente = -1;
        fin = -1;
    }
```

El método insertarDato recibe un argumento String dato e lo inserta en una cola circular. El método primero verifica si la cola está llena usando el método colaEstaLlena() y, si es así, muestra un cuadro de diálogo indicando que la cola está llena y se ha producido un desbordamiento. Si la cola no está llena, luego verifica si la cola está vacía con colaEstaVacia(). Si está vacía, inicializa los índices de frente (frente) y fin (fin) en 0. Si el índice fin fin es igual a maximo, que probablemente representa el tamaño máximo de la cola, reinicia fin a 0 para mantener la naturaleza circular de la cola. Si no se cumple ninguna de las condiciones, simplemente incrementa fin. Finalmente, inserta los datos en el array de la cola circular colaCircular en la posición fin.

```
public void insertarDato(String dato) {
    if (colaEstáLlena()) {
        JOptionPane.showMessageDialog( parentComponent: null,  message: "La cola está llena. Existe desbordamiento.");
    } else {
        if (colaEstáVacía()) { // Si la cola está vacía, inicializar frente y fin a 0
            frente = 0;
            fin = 0;
        } else if (fin == maximo) { // Si fin llega al final del arreglo, retroceder al principio
            fin = 0;
        } else {
            fin++; // Incrementar fin después de la inserción
        }
        colaCircular[fin] = dato; // Insertar el dato en la posición fin
    }
}
```

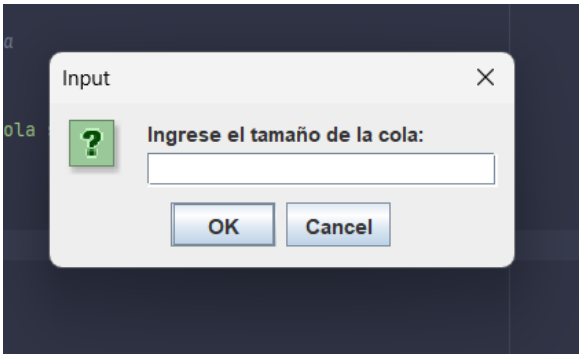
El método eliminarDato parece ser responsable de eliminar un dato de la cola circular. Primero verifica si la cola está vacía con el método colaEstaVacia(). Si la cola está vacía, muestra un cuadro de diálogo indicando que la cola está vacía y que no se puede eliminar nada. Si la cola no está vacía, asigna null al dato en la posición frente para eliminarlo.

Luego, el código verifica si frente es igual a fin. Si son iguales, significa que solo hay un elemento en la cola, por lo que reinicia ambos índices a -1. Si frente es igual a maximo, que probablemente representa el tamaño máximo de la cola, reinicia frente a 0 para mantener la naturaleza circular de la cola. En caso contrario, avanza frente después de la eliminación.

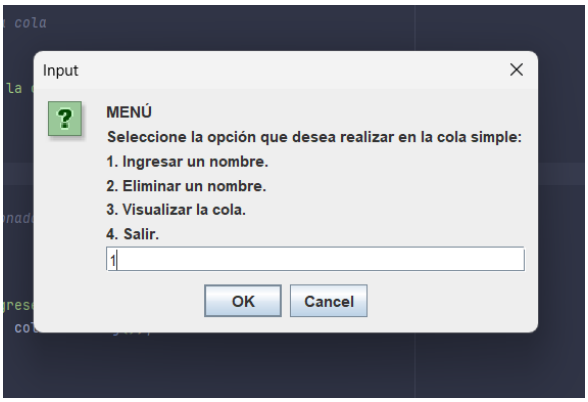
```
public void eliminarDato() {
    if (colaEstáVacía()) {
        JOptionPane.showMessageDialog( parentComponent: null,  message: "La cola está vacía. No se puede eliminar nada.");
    } else {
        colaCircular[frente] = null; // Eliminar el dato en la posición frente
        if (frente == fin) { // Si solo hay un elemento en la cola, reiniciar frente y fin
            frente = -1;
            fin = -1;
        } else if (frente == maximo) { // Si frente llega al final del arreglo, retroceder al principio
            frente = 0;
        } else {
            frente++; // Avanzar frente después de la eliminación
        }
    }
}

// Métodos de apoyo adicionales
```

c. RESULTADOS

Aquí se muestra la ejecución del programa, donde se pide al usuario que ingrese el tamaño que desea de la cola:



Menú principal del programa:



Ingreso de algunos datos:

Aquí se muestra el funcionamiento primordial de una cola circular donde se puede ingresar datos una vez que se llega al final dando la vuelta al arreglo: