package bicola;

/\*\*

\* Métodos para manejar una BiCola

\*

\* Código del libro:

\* Data Structures & Algorithms in Java 2nd Ed., Robert Lafore, 2003.

\*

\* @version 1.00 21/02/11

\*/

class BiCola {

private int MAXIMO;

private long[] arreglo;

private int inicio;

private int fin;

private int nroElementos;

// --------------------------------------------------------------

public BiCola(int t) { // constructor

MAXIMO = t;

arreglo = new long[MAXIMO];

inicio = 1;

fin = 0;

nroElementos = 0;

}

// --------------------------------------------------------------

public void adicionaDer(long j)**----1** { // adiciona un elemento a la derecha de la BiCola

if (!estaLlena()) **----**{

if (fin == MAXIMO - 1) // ¿fin = MAXIMO - 1?

fin = -1;

arreglo[++fin] = j; // incrementa fin y adiciona

nroElementos++; // un elemento mas

}

}

// --------------------------------------------------------------

public void adicionaIzq(long j) { // adiciona un elemento a la izquierda de la BiCola

if (!estaLlena()) {

inicio--;

if (inicio == -1) // ¿inicio = -1?

inicio = MAXIMO-1;

arreglo[inicio] = j; // decrementa inicio y adiciona

nroElementos++; // un elemento mas

}

}

// --------------------------------------------------------------

public long eliminaDer() { // elimina el elementos de la derecha de la BiCola

if (!estaVacia()){

long temp = arreglo[fin--]; // obtiene el valor e decrementa fin

if (fin == -1) // ¿fin = -1?

fin = MAXIMO - 1;

nroElementos--; // un elemento menos

return temp;

} return -1;

}

// --------------------------------------------------------------

public long eliminaIzq() { // elimina el elemento de la izquierda de la BiCola

long valor = Long.MIN\_VALUE;

if (!estaVacia()){

valor = arreglo[inicio++]; // obtiene el valor e incrementa inicio

if (inicio == MAXIMO) // ¿inicio = MAXIMO?

inicio = 0;

nroElementos--; // un elemento menos

}

return valor;

}

// --------------------------------------------------------------

public long peekIzq() { // obtiene el elemento de la izquierda de la BiCola

return arreglo[inicio];

}

// --------------------------------------------------------------

public long peekDer() { // obtiene el elemento de la derecha de la BiCola

return arreglo[fin];

}

// --------------------------------------------------------------

public boolean estaVacia() { // verdad si la BiCola esta vacía

return (nroElementos == 0);

}

// --------------------------------------------------------------

public boolean estaLlena() { // verdad si la BiCola esta llena

return (nroElementos == MAXIMO);

}

// --------------------------------------------------------------

public int tamanio() { // número de elementos de la BiCola

return nroElementos;

}

// --------------------------------------------------------------

public String toString() { // muestra los elementos de la BiCola

String s = "";

for (int i = 0; i <= MAXIMO - 1; i++) {

s = s + "[" + arreglo[i] + "]";

}

s = s + "\n";

return s;

}

// --------------------------------------------------------------

} // fin de la clase BiCola