Homework 4: Stack and Queue

<Stack>

Bài 1. Tạo giao diện StackInterface như sau:

```
public interface StackInterface<E> extends Iterable<E> {
	public void push(E element);
	public E pop();
	public boolean isEmpty();
	public E top();

}
```

1.1 Xây dựng cấu trúc dữ liệu Stack sử dụng mảng, cài đặt giao diện StackInterface đã xây dựng ở trên.

1.2 Xây dựng cấu trúc dữ liệu Stack sử dụng danh sách móc nối, cài đặt giao diện StackInterface đã xây dựng ở trên với lược đồ gợi ý như sau:

```
public class LinkedListStack<E> implements StackInterface<E> {

	class Node {
		E element;
		Node next;
	}

	Node stack = null;

	@Override
	public void push(E element) {
		// Thêm một phần tử vào stack
	}

	@Override
	public E pop() {
		// Lấy một phần tử khỏi stack
		return null;
	}

	@Override
	public boolean isEmpty() {
		// Kiểm tra stack rỗng
		return false;
	}

	@Override
	public E top() {
		// Lấy giá trị phần tử đầu tiên của stack
		return false;
	}

	@Override
	public Iterator<E> iterator() {
		// TODO Auto-generated method stub
		return new StackIterator();
	}
```

```
        class StackIterator implements Iterator<E> {

                private Node currentNode = stack;

                @Override
                public boolean hasNext() {
                        // TODO Auto-generated method stub
                        return currentNode != null;
                }
                @Override
                public E next() {
                        // TODO Auto-generated method stub
                        E data = currentNode.element;
                        currentNode = currentNode.next;
                        return data;
                }
        }
}
```

1.3 Sử dụng stack để làm bài 3.

<Queue>

Bài 2. Xây dựng giao diện QueueInterface như sau:

```
public interface QueueInterface<E> extends Iterable<E> {
      public void enqueue(E element);
      public E dequeue();
      public boolean isEmpty();
}
```

2.1 Xây dựng kiểu dữ liệu Queue sử dụng mảng với lược đồ gợi ý như sau:

```
public class ArrayQueue<E> implements QueueInterface<E> {

      private E[] queue;
      private int n = 0;
      private int top = 0;
      private int count = 0;
      private int default_size = 100;

      public ArrayQueue(int capacity) {
            n = capacity;
            queue = (E[]) new Object[capacity];
      }
      public ArrayQueue() {
            n = default_size;
            queue = (E[]) new Object[default_size];
      }
      @Override
      public void enqueue(E element) {
            // TODO Auto-generated method stub
      }
      @Override
      public E dequeue() {
            // TODO Auto-generated method stub
            return null;
      }
      @Override
      public boolean isEmpty() {
            // TODO Auto-generated method stub
            return false;
      }
```

```
    @Override
    public Iterator<E> iterator() {
        // TODO Auto-generated method stub
        return new ArrayQueueIterator();
    }

    class ArrayQueueIterator implements Iterator<T> {
        private int current = top;
        private int num = 0;
        @Override
        public boolean hasNext() {
            // TODO Auto-generated method stub
            return num < count;
        }
        @Override
        public E next() {
            // TODO Auto-generated method stub
            E data = queue[(current + num) % n];
            num++;
            return data;
        }
    }
}
```

2.2 Xây dựng kiểu dữ liệu Queue sử dụng danh sách móc nối

2.3 Sử dụng queue để làm Bài 4.

Bài 3. Sử dụng stack viết chương trình xét tính hợp lệ về dấu ngoặc của biểu thức:

Ví dụ biểu thức hợp lệ về dấu ngoặc

*(a + b) * (c – d)*

*(10 – 8) / ((2 + 5) * 17)*

Ví dụ biểu thức không hợp lệ về dấu ngoặc

*(a + b) * c – d)*

*(10 – 8 / ((2 + 5) * 17)*

*) u – v) * (m + n)*

Tính giá trị biểu thức nếu hợp lệ về dấu ngoặc

Ví dụ:

Input	Output
*(1 + ((2 + 3) * (4 * 5)))*	*101*
*((50 – ((8 – 4) * (2 + 3))) + (3 * 4))*	*42*

Bài 4. Sử dụng queue kết hợp với stack đã xây dựng ở trên viết chương trình kiểm tra chuỗi Palindrome

PHẦN BÀI TẬP LẬP TRÌNH ỨNG DỤNG
(Nếu chọn làm bài tập này thì không cần làm các bài từ 1 đến 4 ở trên)

Bài 5. Lập trình mô phỏng game
- Tháp Hà nội
- Tìm đường trong mê cung

Bài 6. Giả lập hệ thống điều hành phục vụ khách hàng tại quầy giao dịch viễn thông theo nguyên tắc: có *n* quầy giao dịch, khách đến trước sẽ được phục vụ trước và được sắp xếp vào bàn giao dịch viên ngay khi free. Các giao thức có thể thực hiện trong hệ thống:

- Giao dịch viên kết thúc một phiên phục vụ
- Giao dịch viên nhận khách mới
- Khách đến lấy thứ tự phục vụ
- Khách được xếp vào quầy trống

Những hoạt động thống kê:

- Tổng số khách đã phục vụ của quầy giao dịch
- Số khách, số thời gian phục vụ khách của mỗi giao dịch viên
- ...

Nguyên tắc đánh giá kết quả bài tập lập trình ứng dụng:
- Minh họa được vấn đề lí thuyết đã học trong chương trình (sử dụng stack, queue)
- Khối lượng bài toán được giải quyết (chức năng của chương trình, đồ họa...)