Bài 1:

Tạo giao diện StackInterface như sau:

```
public interface StackInterface<T> extends Iterable<T> {

	public void push(T element);

	public T pop();

	public boolean isEmpty();

}
```

Xây dựng cấu trúc dữ liệu Stack sử dụng danh sách móc nối, cài đặt giao diện StackInterface đã xây dựng ở trên.

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

	class Node {
		T element;
		Node next;
	}

	Node stack = null;

	@Override
	public void push(T element) {
		// TODO Auto-generated method stub

	}

	@Override
	public T pop() {
		// TODO Auto-generated method stub
		return null;
	}

	@Override
	public boolean isEmpty() {
		// TODO Auto-generated method stub
		return false;
	}

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

```
    class StackIterator implements Iterator<T> {

        private Node currentNode = stack;

        @Override
        public boolean hasNext() {
            // TODO Auto-generated method stub
            return currentNode != null;
        }

        @Override
        public T next() {
            // TODO Auto-generated method stub
            T data = currentNode.element;
            currentNode = currentNode.next;
            return data;
        }
    }
}
```

Bài 2:

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

Bài 3:

Xây dựng giao diện QueueInterface như sau:

```
public interface QueueInterface<T> extends Iterable<T> {

    public void enqueue(T element);

    public T dequeue();

    public boolean isEmpty();

}
```

Xây dựng kiểu dữ liệu Queue sử dụng mảng

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

    private T[] 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 = (T[]) new Object[capacity];
    }

    public ArrayQueue() {
        n = default_size;
        queue = (T[]) new Object[default_size];
    }

    @Override
    public void enqueue(T element) {
        // TODO Auto-generated method stub

    }

    @Override
    public T dequeue() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public boolean isEmpty() {
        // TODO Auto-generated method stub
        return false;
    }

    @Override
    public Iterator<T> 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 T next() {
            // TODO Auto-generated method stub
            T data = queue[(current + num) % n];
```

```
                num++;
                return data;
            }

        }

}
```

Bài 4:

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

Bài 5:

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)*

Bài 6:

Tính giá trị biểu thức đầy đủ dấu ngoặc

Ví dụ:

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