PART 1 - Linked List Stack (Implem)

Il vous est demandé d'implémenter l'interface suivante, représentant une pile, en utilisant une liste simplement chainée. Vous devriez avoir au moins un constructeur sans argument, créant une pile vide.

Note: utiliser *java.util.Stack<E>* est interdit!

```
import java.util.EmptyStackException;

public interface Stack<E> {

    public boolean empty();

    public E peek() throws EmptyStackException;

    public E pop() throws EmptyStackException;

    public void push(E item);

}
```

```java
import java.util.EmptyStackException;

public class MyStack<E> implements Stack<E> {

    private Node<E> top;         // the node on the top of the stack
    private int size;         // size of the stack

    // helper linked list class
    private class Node<E> {
        private E item;
        private Node<E> next;

        public Node(E element, Node<E> next) {
            this.item = element;
            this.next = next;
        }
    }

    /**
    * Tests if this stack is empty
    */
    @Override
    public boolean empty() {
        // TODO STUDENT: Implement empty method
    }

    /**
    * Looks at the object at the top of this stack
    * without removing it from the stack
    */
    @Override
    public E peek() throws EmptyStackException {
        // TODO STUDENT: Implement peek method
    }

    /**
    * Removes the object at the top of this stack
    * and returns that object as the value of this function
    */
    @Override
    public E pop() throws EmptyStackException {
        // TODO STUDENT: Implement pop method
    }

    /**
    * Pushes an item onto the top of this stack
    * @param item the item to append
    */
    @Override
    public void push(E item) {
        // TODO STUDENT: Implement push method

    }
}
```

Le projet IntelliJ est disponible ici.

Your answer passed the tests! Your score is 100.0%. [Submission #5d8db2276779dd2b03023ce4]

Your implementation passed all the tests !

Question 1: Implementation de la fonction empty: boolean empty()

```
/**
 * Tests if this stack is empty
 */
@Override
public boolean empty() {
    // TODO STUDENT: Implement empty method
}
```

Copier le contenu de la fonction `public boolean empty()` ci-desssous.

```
if(size==0){
    return true;
}
return false;
```

Question 2: Implementation de la fonction peek: E peek()

```
/**
 * Looks at the object at the top of this stack
 * without removing it from the stack
 */
@Override
public E peek() throws EmptyStackException {
    // TODO STUDENT: Implement peek method
}
```

Copier le contenu de la fonction `public E peek()` ci-desssous.

```
if(size < 1){
    throw new EmptyStackException();
}
return top.item;
```

Information

Author(s)	Simon Hardy, Frédéric Kaczynski
Deadline	01/10/2019 23:55:00
Status	Succeeded
Grade	100.0%
Grading weight	1.0
Attempts	1
Submission limit	No limitation

Submitting as

Gauthier de Moffarts d'H.

Teams management

Project Generator

Download IntelliJ Project

For evaluation

Best submission

Question 3: Implementation de la fonction pop: E pop()

```
/**
 * Removes the object at the top of this stack
 * and returns that object as the value of this function
 */
@Override
public E pop() throws EmptyStackException {
    // TODO STUDENT: Implement pop method
}
```

Copier le contenu de la fonction `public E pop()` ci-desssous.

```
if(size < 1){
    throw new EmptyStackException();
}
E a=top.item;
top.item=null;
top=top.next;
size--;
return a;
```

Question 4: Implementation de la fonction push: void push(E item)

```
/**
 * Pushes an item onto the top of this stack
 * @param item the item to append
 */
@Override
public void push(E item) {
    // TODO STUDENT: Implement push method

}
```

Copier le contenu de la fonction `public push(E item)` ci-desssous.

```
size++;
top=new Node(item,top);
```

Submit

27/09/2019 08:54:31 - 100.0%

Submission history

27/09/2019 08:54:31 - 100.0%