```
package structure;

import structure.exceptions.VectorException;

public class Vector<T> {

	public Vector() {
		init(DEFAULT_VECTOR_SIZE);
	}

	public Vector(int initSize) throws VectorException {
		if (initSize < 0)
			throw new VectorException(new String("init size is
below zero"));
		if (initSize == 0) init(DEFAULT_VECTOR_SIZE);
		else init(initSize);
	}

	// Copy constructor doesn't create copy of elements here.
	// It means that 2 stacks will have links to the same T objects.
	public Vector(Vector<T> v) throws VectorException {
		int size = v.size();
		init(size);
		Object[] temp = new Object[size];
		for (int i = 0; i < size; ++i)
			temp[i] = v.get(i);
		for (int i = size - 1; i >= 0; --i)
		{
			v.set(i, (T)temp[i]);
			set(i, (T)temp[i]);
		}
	}

	public boolean set(int idx, T elem) {
		boolean res = false;
		if (checkIdx(idx)) {
			pool[idx] = elem;
			res = true;
		}
		return res;
	}

	public T get(int idx) throws VectorException {
		if (checkIdx(idx)) return (T)pool[idx];
		else throw new VectorException(new String("wroong index"));
	}
```

```
	public int size() {
		return pool.length;
	}

	public void increaseSize(int newSize) throws VectorException {
		if (newSize <= pool.length)
			throw new VectorException(new String("wrong new
size"));
		Object[] temp = new Object[newSize];
		for (int i = 0; i < pool.length; ++i)
			temp[i] = pool[i];
		pool = temp;
	}

	@Override
	public String toString() {
		String str = new String("Vector(" + pool.length + "):{ ");
		for (int i = 0; i < pool.length; ++i) {
			str += new String("{");
			if (pool[i] == null) str += new String("null");
			else str += new String(pool[i].toString());
			str += new String("} ");
		}
		str += new String("}");
		return str;
	}

	protected void init(int size) {
		pool = new Object[size];
	}

	protected boolean checkIdx(int idx) {
		boolean res = false;
		if (pool != null) {
			if (idx >= 0 && idx < pool.length)
				res = true;
		}
		return res;
	}

	protected Object[] pool; // stores T[] array

	private final int DEFAULT_VECTOR_SIZE = 10;

}
```