CSC 212 Tutorial #1
Generics

Problem 1

Complete the method *search* below that takes as input a generic array *data* of size n and searches for the element e. If e is found, the index of its first occurrence is returned, otherwise -1 is returned.

```
public class Utils {
    public static <T> int search(T[] data, int n, T e) {
        ...
    }
}
```

Problem 2

The following is the specification of a data structure called *GenericArray*:

- **Domain**:
  - Structure: linear.
  - Data type: generic.
- **Operations**: All operations will be done on a *GenericArray arr* of size n.
  - Procedure get(int i, T e). **Requires**: $0 \leq i < n$. **Results**: e is set to the element of *arr* at position i.
  - Procedure set(int i, T e). **Requires**: $0 \leq i < n$. **Results**: the element of *arr* at position i is set to e.

1. Complete the implementation of *GenericArray* below:

```
public class GenericArray<T> {
    ...

    public GenericArray(int n) {
        ...
    }
    public T get(int i) {
        ...
    }

    public void set(int i, T val) {
        ...
    }
}
```

2. Consider the class *Box*:

```
public class Box<T> {
    private T data;

    public Box(T data) {
        this.data = data;
    }

    public T get() {
        return data;
    }

    public void update(T data) {
        this.data = data;
    }
}
```

and a method that uses this class by creating an array of *Box<String>*:

```
public class ArrayOfBox {
    public static void main(String[] args) {
        Box<String>[] b = new Box<String>[3];
        b[0] = new Box<String>("A");
        b[0].update("B");
    }
}
```

What happens when you compile this code. Use the class *GenericArray* to solve this problem.