3. Answer the following questions for the method `search()` below:

```
public static int search (List list, Object element)
// Effects: if list or element is null throw NullPointerException
//   else if element is in the list, return an index
//   of element in the list; else return -1
//   for example, search ([3,3,1], 3) = either 0 or 1
//       search ([1,7,5], 2) = -1
```

Base your answer on the following characteristic partitioning:

```
Characteristic: Location of element in list
       Block 1: element is first entry in list
       Block 2: element is last entry in list
       Block 3: element is in some position other than first or last
```

(a) "Location of element in list" fails the disjointness property. Give an example that illustrates this.

(b) "Location of element in list" fails the completeness property. Give an example that illustrates this.

(c) Supply one or more new partitions that capture the intent of "Location of element in list" but do not suffer from completeness or disjointness problems.

4. Derive input space partitioning test inputs for the `GenericStack` class with the following method signatures:

- `public GenericStack ();`
- `public void push (Object X);`
- `public Object pop ();`
- `public boolean isEmpty ();`

Assume the usual semantics for the `GenericStack`. Try to keep your partitioning simple and choose a small number of partitions and blocks.

(a) List all of the input variables, including the state variables.
(b) Define characteristics of the input variables. Make sure you cover all input variables.
(c) Partition the characteristics into blocks.
(d) Define values for each block.