Aufgabe 6 (Stacks)

Gegeben sei die Implementierung eines Stacks ganzer Zahlen mit folgender Schnittstelle:

```
3  import java.util.Stack;
4
5  /**
6   * Um schnell einen lauffähigen Stack zu bekommen, verwenden wir den Stack aus
7   * der Java Collection.
8   */
9  public class IntStack {
10   private Stack<Integer> stack = new Stack<Integer>();
11
12   // legt Element i auf den Stack
13   public void push(int i) {
14     stack.push(i);
15
16   }
17
18   // gibt oberstes Element vom Stack
19   public int pop() {
20     return stack.pop();
21   }
22
23   // fragt ab ob Stack leer ist
24   public boolean isEmpty() {
25     return stack.empty();
26   }
27 }
```

Betrachten Sie nun die Realisierung der folgenden Datenstruktur `Mystery`, die zwei Stacks benutzt.

```
3  public class Mystery {
4    private IntStack a = new IntStack();
5    private IntStack b = new IntStack();
6
7    public void foo(int item) {
8      a.push(item);
9    }
10
11   public int bar() {
12     if (b.isEmpty()) {
13       while (!a.isEmpty()) {
14         b.push(a.pop());
15       }
16     }
17     return b.pop();
18   }
```

(a) Skizzieren Sie nach jedem Methodenaufruf der im folgenden angegebenen Befehlssequenz den Zustand der beiden Stacks eines Objekts `m` der Klasse `Mystery`. Geben Sie zudem bei jedem Aufruf der Methode `bar` an, welchen Wert diese zurückliefert.

```
21       Mystery m = new Mystery ();
22       m.foo(3);
23       m.foo(5);
24       m.foo(4);
```

```
25      m.bar();
26      m.foo(7);
27      m.bar();
28      m.foo(2);
29      m.bar();
30      m.bar();
```

(b) Sei n die Anzahl der in einem Objekt der Klasse `Mystery` gespeicherten Werte. Im folgenden wird gefragt, wieviele Aufrufe von Operationen der Klasse `IntStack` einzelne Aufrufe von Methoden der Klasse `Mystery` verursachen. Begründen Sie jeweils Ihre Antwort.

(i) Wie viele Aufrufe von Operationen der Klasse `IntStack` verursacht die Methode `foo(x)` im besten Fall?

(ii) Wie viele Aufrufe von Operationen der Klasse `IntStack` verursacht die Methode `foo(x)` im schlechtesten Fall?

(iii) Wie viele Aufrufe von Operationen der Klasse `IntStack` verursacht die Methode `bar()` im besten Fall?

(iv) Wie viele Aufrufe von Operationen der Klasse `IntStack` verursacht die Methode `bar()` im schlechtesten Fall?

(c) Welche allgemeinen Eigenschaften werden durch die Methoden `foo` und `bar` realisiert? Unter welchem Namen ist diese Datenstruktur allgemein bekannt?