Aufgabe 1: Walkthrough I

Führen Sie in Ihrem Lernteam einen Code-Walkthrough durch. Verwenden Sie dazu eine von Ihnen bereits bearbeitete Programmieraufgabe Ihrer Wahl aus OOP5 oder aus OOP6.

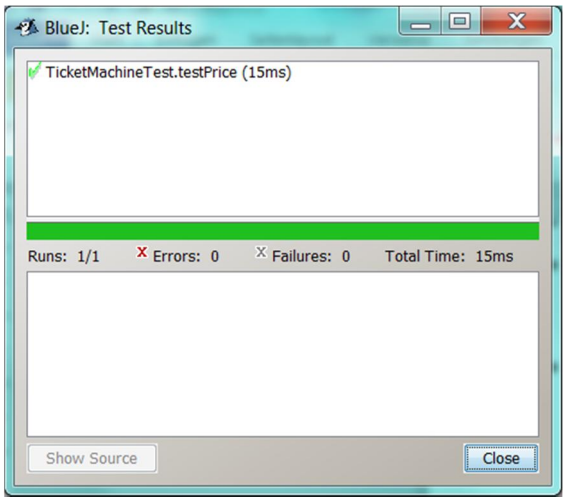
Erstellen Sie ein Kurzprotokoll und berichten Sie anlässlich des Lernteamcoachings über Ihre Erfahrungen.

Code wird angeschaut, wird nicht im Detail überprüft. Einzelne Bereiche werden ggf. vertieft angeschaut, jedoch nicht alles und jedes verifiziert.

Aufgabe 2: Unit Test

Führen Sie mit JUnit einen Test der "TicketMachine2" aus dem BlueJ-Buch durch.

```
public void testPrice()
{
    TicketMachine t1 = new TicketMachine(5);
    assertEquals(5, t1.getPrice());
}
```



Aufgabe 3: Ackermann

```
public static int ack(int n, int m)
{
  if(n==0) {
    return m+1;
  }
  else {
    if(m==0) {
      return ack(n-1, 1);
    }
    else {
      return ack(n-1, ack(n, m-1));
    }
  }
}
```

a) Bestimmen Sie die Rekursionsbasis und die Rekursionsvorschrift.

Rekursionsbasis: *n == 0*

Rekursionsvorschrift: *ack(n-1, 1)*

ack(n-1, ack (n, m-1))

b) Auf Papier, also ohne Computer: Wie oft wird nach dem Aufruf von ack(1, 3) die Methode insgesamt aufgerufen?

0	*ack(n, m-1)*	*1, 3*	
1	*-ack(n, m-1)*	*1, 2*	
2	*--ack(n, m-1)*	*1, 1*	
3	*---ack(n, m-1)*	*1, 0*	
4	*---ack(n-1, 1)*	*0, 1*	*return 2*
5	*--ack(n-1, [2])*	*0, 2*	*return 3*
6	*-ack(n-1, [3])*	*0, 3*	*return 4*
7	*ack(n-1, [4])*	*0, 4*	*return 5*

Die Methode ack() wird siebenmal aufgerufen.

c) Mit dem Computer: Überprüfen Sie Ihr obiges Resultat (z.B. mit Hilfe des Debuggers).

ack(1, 3) => Count: 7

d) Mit dem Computer: Wie viele Aufrufe sind es für ack(3, 3). Dazu müssen Sie in obigem Source-Code noch das Zählen der Methodenaufrufe "einbauen". (Für Ihr Experimentieren mit allzu grossen Parameter -Werten übernehmen wir aber keine Verantwortung!)

ack(3, 3) => Count: 2‘431

ack(3, 5) => Count: 42‘437
ack(3, 6) => Count: 172‘232
ack(3, 10) => Count: 44‘698‘324
ack(3, 11) => Count: 178‘875‘095

```
/**
 * Write a description of class Ackermann here.
 *
 * @author Christian Bontekoe & Felix Rohrer
 * @version 1.0
 */
public class Ackermann
{
    // instance variables - replace the example below with your own
    private int count;

    /**
     * Constructor for objects of class Ackermann
     */
    public Ackermann()
    {
        // initialise instance variables
        count = -1;
```

```
    }

    public int ack(int n, int m)
    {
        count++;
        if(n==0) {
            return m+1;
        }
        else {
            if(m==0) {
                return ack(n-1, 1);
            }
            else {
                return ack(n-1, ack(n, m-1));
            }
        }
    }

    public void testAck(int n, int m)
    {
      count = -1;
      ack(n, m);
      System.out.println("ack(" + n + ", " + m +") => Count: " + count);
    }
}
```