Nachdenkzettel: Threads

Aufgabe 1

Parallel

Aufgabe 2

No, it doesn't solve the problem.

Aufgabe 3

- static variables → yes
- attributs in objects → yes
- stack variables → yes
- heap variables → no

Aufgabe 4

```
class Foo {
private ArrayList aL = new ArrayList();
public int elements;
private String firstName;
private String lastName;
public Foo () {};
public void synchronized setfirstName(String fname) {
firstName = fname;
elements++;
}
public void synchronized setlastName(String lname) {
lastName = lname;
elements++;
}
public synchronized ArrayList getList () {
return aL;
}
}
```

Aufgabe 5

It is problematic, because you can destroy something during the process. You should wait until the program is finished or you let the thread sleep.

Aufgabe 6

It depends on the optimization. If the application is optimized for multi-core, it is faster there.

Aufgabe 8

Infinite loops, because flag fulfill its own condition.

Aufgabe 9

1. Deadlocks, because the threads never end
2. Doing Y

   Doing X

   Doing Y

   Doing X

   Doing Y

   Doing X

   …