Amal Bolakhrif (7423946), Beyza Karaca (7422370), Lena Igoumrane (7424588)

Homework 5

HW4 Task 4 ist auf der ersten Seite des Dokuments unter Task 1. Die Aufgabe wurde nicht vergessen!!

1) Professional Tone = no sarcastic or dismissive language like "Basic programming skill knowledge i)
   → stay neutral
2. Provide concrete examples or suggestions for refactoring the if statements and loops
   → comments like "try to make it faster" are not helpfull
3. no speculative comments like "there might be a security risk"
   → rather suggest techniques to check
4. no generalizations like "the code could be more efficient and readable"
   → point out examples
5. ask clarifying questions rather than making assumptions
6. more positive feedback
7. Organize feedback → in clear sections
8. refer to specific strategy pattern then just of "consider using a strategy pattern to simplify the if statement"

Homework 05

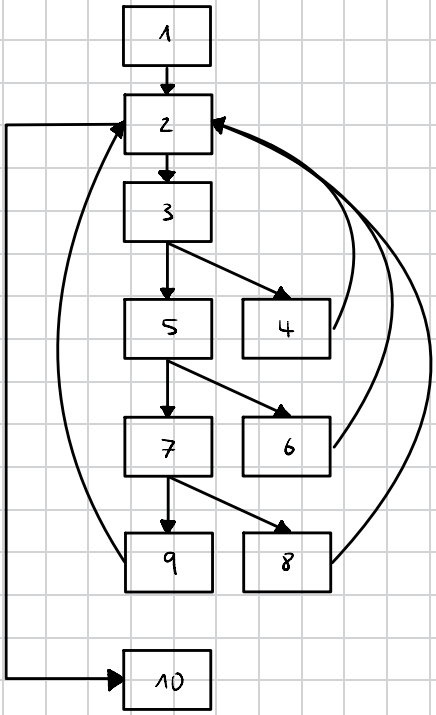
Task 2

	Test Cases	TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10
total Students (a)	a < 0	X									
	a ≥ 0		X	X	X	X	X	X	max	X	X
group Size (b)	b < 0			X							
	b > 0	X	X			X	X	X	X	max	X
	b = 0				X						
available Group (c)	m = 0					X					
	m > 0	X	X	X	X		X	X	X	X	max
	exception	X		X	X	X					
	input a	-5	0	15	15	20	20	22	maxint	20	20
	input b	10	10	-3	0	5	5	5	5	maxint	5
	expected output	exception	0	exception	exception	exception	0	1	maxint -5	0	0
	result	Exception	0	Exception	Exception	exception	0	1	maxint	0	0
		class	boundary	class	boundary	boundary	class	boundary	boundary	boundary	boundary

Exercise 3

The code:

```
1  public String attemptAssignToGroup(
2          List<Student> students,
3          Group group
4      ) {
5
6      List<Student> assignedStudents = new ArrayList<>();                  // 1
7
8      for(int i = 0; i < students.size(); i++) {                           // 2
9          if(students.get(i) == null                                       // 3
10                 || student.get(i).getID() == null) {
11             System.out.println("Invalid student or student ID");        // 4
12             continue;
13         }
14         if(assignedStudents.contains(student)) {                         // 5
15             System.out.println("Student already assigned");              // 6
16             continue;
17         }
18         if(assignedStudents.size() >= group.getCapacity()) {             // 7
19             System.out.println("Group is full");                         // 8
20             continue;
21         }
22
23         // All checks passed, add student to group
24         assignedStudents.add(student);                                   // 9
25     }
26
27     return assignedStudents;                                             // 10
28 }
```



Statement Coverage:

Test Case 1: `testInvalidStudentId()`

- `group.capacity = 5, students = [null]`
- (executes node 1, 2, 3, 4, and 10) (siehe ✓)
- covers line 6, 8, 9-10* (I'll consider l.9 and 10 as one), 11, 12 and 27;

Test Case 2: `testSuccessfulAssignment()`

- `group.capacity = 5, students = [Student ("Sarah Madison", 928439)]`
- (executes node 1, 2, 3, 5, 7, 9 and 10) (siehe ✓)
- covers line 6, 8, 9-10, 14, 18, 24, 27

(Node 6 and 8 are never executed: 8/10 = 0,8 = 80% (considering nodes))

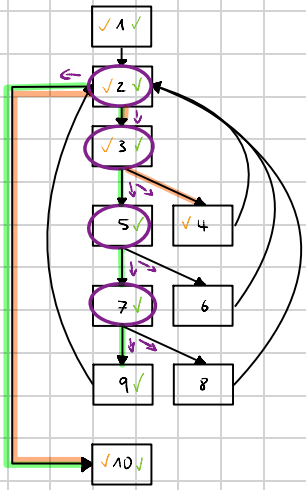
Statement Coverage = lines covered / lines total = 9/13 = 0,692 = 69,2%

Branch Coverage:

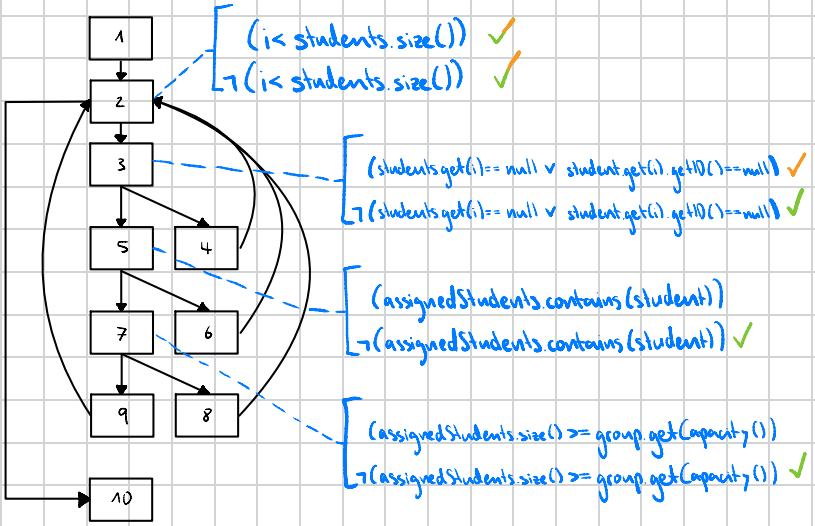
Decision 1: (for): 2 branches (enter/ continue loop or exit)
Decision 2: (if): 2 branches (true or false)
Decision 3: (if): 2 branches (true or false)
Decision 4: (if): 2 branches (true or false)
Total branches: 4 x 2 = 8 branches

Test Case 1: `testInvalidStudentId()` covers 3 out of 8 possible branches
Test Case 2: `testSuccessfulAssignment()` covers 5 out of 8 possible branches
(both share 2 same branches)

Branch Coverage = decision outcomes covered/ decision outcomes total = 6/8 = 75%



Condition Coverage:
(= condition outcomes covered/ condition outcomes total)
Test Case 1: `testInvalidStudentId()`: 3/8
Test Case 2: `testSuccessfulAssignment()`: 5/8
Together: **6/8 = 75%**



Path Coverage:
cases:
Zero iterations: for-loop is never entered (`students.size() == 0`).
One iteration: for-loop is entered once (`students.size() == 1`).
(Multiple iterations: limit to key paths (from one iteration)
possible cases with single loop iteration:
Path 1: 1 → 2 → 3 → 4 → 2 → 10
Path 2: 1 → 2 → 3 → 5 → 6 → 2 → 10
Path 3: 1 → 2 → 3 → 5 → 7 → 8 → 2 → 10
Path 4: 1 → 2 → 3 → 5 → 7 → 9 → 2 → 10
case where loop not executed at all:
Path 5: 1 → 2 → 3 → 5 → 7 → 8 → 2 → 10

Test Case 1: `testInvalidStudentId()`
- **Executes Path 1** (since `students = [null]` means `students.size() == 1` and loop is entered and first if-condition true)

Test Case 2: `testSuccessfulAssignment()`
- **Executes Path 4** (since `students = [Student("Sarah Madison", 928439)]` means loop is entered (`students.size() == 1`) and all presented if-conditions are false and student is added)

Path Coverage = (Covered Paths / Total Practical Paths) = 2/5 = 0,4 = 40%