UNIVERSITÄT
BERN

P2 - Exam Preparation

Terminology

What is the difference between Stack and Heap?

Terminology

Why do “God classes” and “Data classes” often occur together?

Terminology

When should you call super() in a constructor and why?

Terminology

What is Iterative Development, and how does it differ from the Waterfall model?

Terminology

What are models, view and controllers?
What are the advantages of using the Model-View-Controller pattern?

Patterns

Explain the observer pattern on an example use case of your choice. Include the following in your answer:

1. Provide example code.
2. Provide a UML diagram of the classes involved.
3. Give one advantage and one disadvantage of using the Observer pattern to implement a GUI. Less than 100 words.

Patterns

You should be able do describe and use all the pattern from the lecture and from the lab.

- Adapter
- Proxy
- Template Method
- Composite
- Observer
- Null Object

- Factory Method
- State
- Visitor
- Abstract Factory
- Builder
- Chain of Responsability

Code Comments

Critique the following code comments:

```
/**
* The <i>Algorithm</i> defines how a value for a file is computed.
* It must be sure that multiple calls for the same file results in the
* same value.
* The implementing class should implement a useful toString() method.
*
* @version 2003-09-13
* @since Ant 1.6
*/
public interface Algorithm {
    /**
    * Get the value for a file.
    * @param file File object for which the value should be evaluated.
    * @return The value for that file
    */
    String getValue(File file);
}
```

Testing

Write a Junit test that verifies that line 10 works as expected.

```
01. public class Spreadsheet {
02.    private int[][] contents;
03.    private int rows;
04.    private int cols;
   ....
05. public void setCellValue(int row, int col, int value) {
06.    if (row < 0 || row > this.rows-1) {
07.       throw new IllegalArgumentException();
08.    }
09.    if (col < 0 || col > this.cols-1) {
10.       throw new IllegalArgumentException();
11.    }
12.    this.contents[row][col] = value;
13. }
   ....
```

Design principles

Consider the following code snippets of a board game. The unit test creates a piece, puts it on a board and then sees whether the piece has advanced by one field after being asked to proceed by one field.

```
@Given("#game")
public void shouldMoveIntoPlayWithFiveOnly(Game game) {
    Square out = game.getCurrentPlayer().getStartField();
    playOneGameStep(game, 1);
    assertFalse(out.occupied());
    playOneGameStep(game, 5);
    assertTrue(out.occupied());
}
private void playOneGameStep(Game game, int steps) {!
    game.getDie().setNextRoll(steps);
    game.playNextTurn();
}
```

1. Does this code snippet violate the law of Demeter? If so, where?!
2. Do you see a conflict between the law of Demeter and responsibility-driven design in this example? Justify your opinion. Outline the law of Demeter and responsibility-driven design in your answer. Less than 130 words.

Smalltalk

Explain what the following Smalltalk code does in 100 words.

```
rows: rows columns: columns tabulate: aBlock
    |a i|
    a := Array new: rows*columns.
    i := 0.
    1 to: rows do: [:row |
        1 to: columns do: [:column |
            a at: (i := i+1) put: (aBlock value: row value: column)]].
    ^a
```

Terminology

What is the equivalent of a static method in Smalltalk?