Logistics

- **Paper review** is due on 10/06 4pm (strict)!

- **Homework 2** is due on 10/07 (extended)

- **Paper selection** between 10/10 9am and 10/11 11:55pm
Recap: what is program analysis?

- (Automatically) analyze the behavior of a program
  - Optimize the program or
  - check program’s behavior (against its specification)
- Concerned with properties such as
  - Correctness
  - Safety
  - Liveness
  - Performance
- Can be static or dynamic or a combination of both
Recap: code review/inspection

Different types of reviews

● Code/design review
● Informal walkthrough
● Formal inspection
Recap: code review/inspection

Different types of reviews
- Code/design review
- Informal walkthrough
- Formal inspection

Pros
- Can be applied at any step in the development process
- Improves confidence and communication

Cons
- Time-consuming
- Mostly informal
- Not repeatable
Today

Research paper reading/presentation
  ● Logistics and overview of research papers
  ● How to choose a paper?
  ● How to present a paper?

MVC revisited
  ● Proper decoupling
  ● The Observer pattern
  ● Q & A
Research papers: logistics

- 8 papers in 2 research areas
  - Software testing: 4 papers (October)
  - Software debugging: 4 papers (December)

- Papers are grouped by 4 high-level topics
  - Metrics for test effectiveness
  - Automated test generation
  - Fault localization
  - Automated debugging and program repair
Research papers: logistics cont.

- Each paper has a fixed presentation date
- **Paper assignments** through Moodle *(10/10, 9am)*
  *(first come, first serve)*

- **620 students**
  - Review 1 paper
  - Present 1 paper (in addition to the reviewed paper)

- **520 students**
  - Review 1 paper
  - *Optional:* present 1 paper
Research papers: logistics cont.

- Paper reviews
  - 4 students per group
  - Paper review is a group submission

- Paper presentations
  - 2 or 3 students per group (depends on the paper)
  - Paper presentation is a group presentation
  - 20 minute presentation
How to choose a paper?

1. Take a brief look at each paper
2. Read the abstract (and conclusions)
3. Rank the papers based on interests
4. Sign up for a paper (first come, first serve)
How to present a paper?

1. Give a brief overview of the area/domain and problem
2. Outline research questions and contributions
3. Describe the approach/methodology
4. Highlight key results (evaluation)
5. Outline and discuss conclusions
6. Discuss limitations and directions for future work
How to present a paper?

1. Give a brief overview of the area/domain and problem
2. Outline research questions and contributions
3. Describe the approach/methodology
4. Highlight key results (evaluation)
5. Outline and discuss conclusions
6. Discuss limitations and directions for future work
7. Outline questions or controversial statements for the classroom discussion.
Software testing vs. software debugging

```java
double avg(double[] nums) {
    int n = nums.length;
    double sum = 0;

    int i = 0;
    while (i<n) {
        sum = sum + nums[i];
        i = i + 1;
    }

    double avg = sum * n;
    return avg;
}
```
Software testing vs. software debugging

**Testing: is there a bug?**

```java
@Test
public void testAvg() {
    double nums = 
        new double[]{1.0, 2.0, 3.0});
    double actual = Math.avg(nums);
    double expected = 2.0;
    assertEquals(expected, actual, EPS);
}
```

```java
double avg(double[] nums) {
    int n = nums.length;
    double sum = 0;
    int i = 0;
    while (i<n) {
        sum = sum + nums[i];
        i = i + 1;
    }
    double avg = sum * n;
    return avg;
}
```
Software testing vs. software debugging

Testing: is there a bug?

```java
testAvg() {
    double[] nums = {1.0, 2.0, 3.0};
    double actual = Math.avg(nums);
    double expected = 2.0;
    assertEquals(expected, actual, EPS);
}
```

testAvg failed: 2.0 != 18.0
Software testing vs. software debugging

Testing: is there a bug?
@Test
public void testAvg() {
    double nums =
        new double[]{1.0, 2.0, 3.0});
    double actual = Math.avg(nums);
    double expected = 2.0;
    assertEquals(expected, actual, EPS);
}

testAvg failed: 2.0 != 18.0

Debugging: where is the bug?
how to fix the bug?

double avg(double[] nums) {
    int n = nums.length;
    double sum = 0;
    int i = 0;
    while (i<n) {
        sum = sum + nums[i];
        i = i + 1;
    }
    double avg = sum * n;
    return avg;
}
Software testing vs. software debugging

```java
1 double avg(double[] nums) {
2     int n = nums.length;
3     double sum = 0;
4
5     int i = 0;
6     while (i<n) {
7         sum = sum + nums[i];
8         i = i + 1;
9     }
10
11    double avg = sum * n;
12    return avg;
13 }
```

Testing: is there a bug?
@Test
class Main {
    public void testAvg() {
        double nums =
            new double[]{1.0, 2.0, 3.0});
        double actual = Math.avg(nums);
        double expected = 2.0;
        assertEquals(expected, actual, EPS);
    }
}

testAvg failed: 2.0 != 18.0

Debugging: where is the bug?
how to fix the bug?
Selected papers: software testing

Metrics for test effectiveness -- 10/18

- The Major mutation framework. ISSTA 2014.
- Are mutants a valid substitute for real faults in software testing? FSE 2014.
- Can testedness effectively be measured? FSE 2016.

Opportunity for 520 students!
Selected papers: software testing

**Metrics for test effectiveness -- 10/18**
- The Major mutation framework. ISSTA 2014.
- Are mutants a valid substitute for real faults in software testing? FSE 2014.
- Can testedness effectively be measured? FSE 2016.

**Automated test generation -- 10/25**
- Do automatically generated unit tests find real faults? ASE 2015.

Opportunity for 520 students!
Selected papers: software debugging

Fault localization -- 11/29

- Defects4J. ISSTA 2014.
- Are automated debugging techniques actually helping programmers? ISSTA 2011.

Opportunity for 520 students!
Selected papers: software debugging

Fault localization -- 11/29

● Defects4J. ISSTA 2014.
● Are automated debugging techniques actually helping programmers? ISSTA 2011.

Automated debugging/program repair -- 12/06

● Isolating Cause-Effect Chains from Computer Programs. FSE 2002.
● Improving delta debugging. ICSE 2013.
● Anti-patterns in search-based program repair. FSE 2016.

Opportunity for 520 students!
MVC revisited

The MVC architecture and the Observer pattern
MVC revisited

The MVC architecture and the Observer pattern

The controller does not update the view!
### MVC revisited: weather station

<table>
<thead>
<tr>
<th>Current</th>
<th>30 day history</th>
</tr>
</thead>
<tbody>
<tr>
<td>25° F</td>
<td></td>
</tr>
<tr>
<td>-3.9° C</td>
<td>min: 20° F</td>
</tr>
<tr>
<td></td>
<td>max: 35° F</td>
</tr>
</tbody>
</table>

- **Temp. sensor**
- **Reset history button**
- **Reset**
MVC revisited: weather station

- **25° F**
- **-3.9° C**
  - min: 20° F
  - max: 35° F

Client

- Sees
- Uses

**Reset**

**Temp. sensor**

**Reset history button**

09/01, 12°
09/02, 14°
...
MVC revisited

See code examples (online)