

# CSE 403

## Software Engineering

### **Testing & CI**

# Today

- Software testing 101
- Testing best practices
- Continuous Integration
- Live demo and Q & A

# Software testing 101

# Software testing vs. software debugging

```
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 }
```

# Software testing vs. software debugging

```
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**

```
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);  
}
```

# Software testing vs. software debugging

```
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?

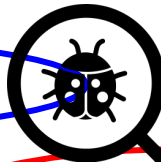
@Test

```
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

# Software testing vs. software debugging

```
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

```
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);  
}
```

**FAIL**

testAvg failed: 2.0 != 18.0

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

# Testing best practices



# Testing best practices: motivating example

## Average of the absolute values of an array of doubles

```
public double avgAbs(double ... numbers) {  
  
    // We expect the array to be non-null and non-empty  
    if (numbers == null || numbers.length == 0) {  
        throw new IllegalArgumentException("Array numbers must not be null or empty!");  
    }  
  
    double sum = 0;  
    for (int i=0; i<numbers.length; ++i) {  
        double d = numbers[i];  
        if (d < 0) {  
            sum -= d;  
        } else {  
            sum += d;  
        }  
    }  
  
    return sum/numbers.length;  
}
```

**What tests should we write for this method?**

# Testing best practices: motivating example

## Compare two values

```
public class Comp implements Comparable<Comp> {
    private int number;
    public Comp(int number) {this.number = number;}

    @Override
    public int compareTo(Comp other) {
        if (other.number == this.number) return 0;
        return this.number < other.number ? -1 : 1;
    }
}

public class CompTest {
    @Test
    public void testSmaller() {
        Comp c1 = new Comp(10);
        Comp c2 = new Comp(20);
        assertEquals(c1.compareTo(c2), -1);
    }
}
```

**Is this a desirable and effective test?**

# Live example: test automation

## Testing best practices

- Test atomicity
- Table-based testing
- Parameterized unit tests

# Continuous Integration

# CI/CD: What's the difference?

## Continuous Integration (CI)

- Integrates code into a shared repository.
- Builds/tests each change automatically.
- Complements local developer workflows (subset of tests vs. all tests).

## Continuous Deployment (CD)

- Builds on top of CI.
- Software can be deployed at any time.
- Automatically pushes changes to production.

**403 focuses on establishing good CI practices.**

# **Live example: CI in action**

## CI examples

- Travis CI
- GitHub Actions

# Types of software tests

# Unit testing, integration testing, system testing

## System testing (E2E)

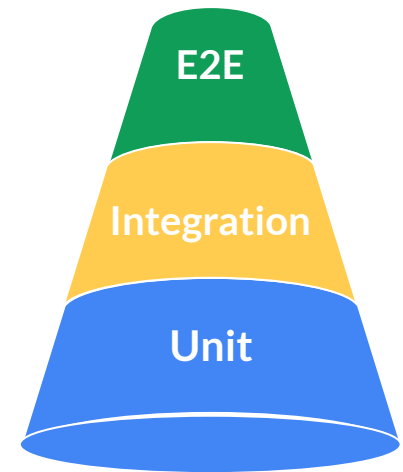
- Does the system work as a whole?

## Integration testing

- Do the units work when put together?

## Unit testing

- Does each unit work as specified?



**This makes sense conceptually, but ...**

**What exactly is a unit?**

**Is it possible and desirable to test units in isolation?**