Lecture 2: class activities

prof. dr. Irma Ravkic

1 Activity 2.1: analysis

Analyze two code segments below. Try to explain their function in your own words. Now, can you compare them in terms of efficiency? Which code is more efficient? (Hint: ask yourself how these code segments would perform if the name you're searching is the first in the list, or if it is last in the list.)

Listing 1: Code # 1

```
boolean isRegistered(String searchName) {
  for (String name: registeredList) {
      if (name.equals(searchName))
          return true;
  }
  return false;
}
```

Listing 2: Code #2

```
 void isRegistered(String searchName) {
      boolean found = false;
      for (String name: registeredList) {
          if (name.equals(searchName))
              found = true;
      }
      if (found)
          return true;
      else
          return false;
}
```

2 Activity 2.2: analyze the complexity of a program

Determine the big-O complexity of the program in Listing 3.

Listing 3: Code #3

```java
public static void someFunction(int[] items){
        System.out.println(items[0]);
        int middleIndex = items.length / 2;
        int index = 0;
        while(index < middleIndex) {
            System.out.println(items[index]);
            index++;
        }
        for(int i = 0; i < 100; i++){
            System.out.println("hi");
        }
```

3 Activity 2.3: analysis...again

Determine the number of steps needed to execute the program below (also count assignments). Then determine the order of magnitude or big-O.

Listing 4: Code # 4

```java
int a=5
int b=6
int c=10
for(int i = 0; i < n; i++){
   for(int j = 0; j < n; j++){
      int x = i * i;
      int y = j * j;
      int z = i * j;
   }
}
for(int k = 0; k < n; k++){
   int w = a*k + 45;
   int v = b*b;
}
int d = 33;
```

4 Activity 2.4: proove complexity mathematically

Show that f(n) = 3n + 7 is O(n)

5 Activity 2.5: minimum

Write two Java functions to find the minimum number in a list/array. The first function should compare each number to every other number on the list which is $O(n^2)$. The second function should be linear $O(n)$.

6 Activity 2.6: sum digits in a number big O

The following code sums the digits in a number. What is its big O time?

Listing 5: Code # 5

```java
int sumDigits(int n) {
    int sum = 0;
    while (n > 0) {
        sum += n % 10;
        n /= 10;
    }
    return sum;
}
```

7 Activity 2.6: be smart about building your strings

In your activity folder you will find a Java project called *StringDemo*. Load the project into Eclipse (or any other IDE you are using). Analyze the code in front of you and try to determine what it is doing. Then think about the implications of the analysis. Does it make you smarter about how to use Strings in your future Java projects?

8 Activity 2.7: could it be linear?

For each of the following problems, decide whether it can be solved by a linear algorithm and explain briefly why or why not.

- Compute the sum of a list containing N random integers.
- Determining if duplicate numbers occur in a list of N random integers.