(Prepared by Avraham Shalev, [avraham.shalev@post.idc.ac.il](mailto:avraham.shalev@post.idc.ac.il.com))

**Functions workshop**

The aim of this workshop is to help students feel more comfortable with using and writing functions, while focusing string operations.

**Methodology**

1. Recap (5-10 mins)
2. Strings (5 mins)
   1. Reminder
   2. Warm-up question
3. Functions (10-15 mins)
   1. Reminder
   2. Warm-up questions
4. Genes (20-25 mins)
5. HTML (10 mins)

**Recap (Functions, Loops, Strings)** (5-10 mins)

1. Function signature
   1. ~~private\public~~
      1. private unless you have a good reason
   2. ~~static\not~~
      1. instance-relative or global method?
   3. return type?
      1. Consider all edge cases, what is the most suitable type?
   4. Arguments count
      1. Think generally as much as you can. This method might serve other use-cases in the future
   5. Arguments type?
      1. Consider all edge cases, what is the most suitable type?
   6. Arguments name?
      1. Very,very,VERY important! Choose wisely
   7. Function overloading (same name for different functions in the same class) can be done by different signatures (params\return type, ..)
   8. Our target is that one can tell what this function does only by it’s signature
      1. Int tempFunc(int a, int b) VS int sum(int num1, int num2)
2. Loops
   1. for(int i=0;i<10;i++)
      1. Pattern - for(A;B;C).
         1. A is for one-time initialization. B is a boolean condition that the loop checks before each iteration. C is a command to run after each iteration of the loop.
         2. Each of them (A\B\C) is optional (it’s valid to run “for(;;)” )
   2. While(i<10)
      1. Only one aspect – boolean condition.
   3. Do { ….. } while (i<10)
      1. Same as while, except that the first iteration is happening before the boolean condition is checked.
3. Strings (4 main functions and their overloads)
   1. s.length()
   2. s.charAt(int index)
   3. s.indexOf(char c)
   4. s.substring(int start, int end)

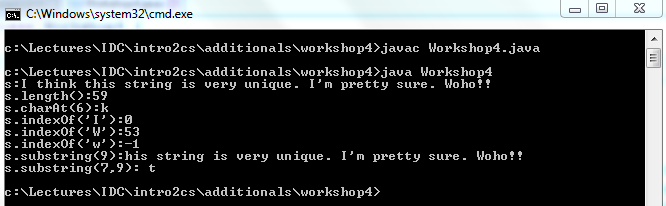
**Strings** (5 mins)

All the functions we’ll use today will handle Strings and use their associated functions.

For a String ‘s’, the functions we will care about are:

* s.length()
* s.charAt(int index)
* s.indexOf(char c)
* s.indexOf(String sub)
* s.substring(int start)
* s.substring(int start, int end)

Demonstrate the use of these functions with a test String. Below is one possible example.



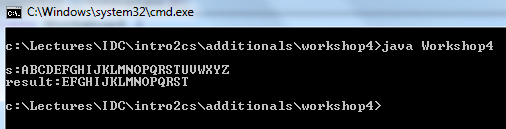
* note that s.indexOf(‘W’) != s.indexOf(‘w’)
* note that s.indexOf(‘I’)=0, which is the first occurrence (there is an overloaded method for occurrence n)
* note when s1=”123” and s2=”123”:
  + s1.equals(s2) is true
  + s1==s2 is false (“==” compares Strings by reference and not by value)

**Warm-up Question**

Ask students to solve the following question. Start with the String

s = "ABCDEFGHIJKLMNOPQRSTUVWXYZ". From this String, construct a new String which contains all letters between ‘E’ and ‘T’ (including both). Students may only use String functions.

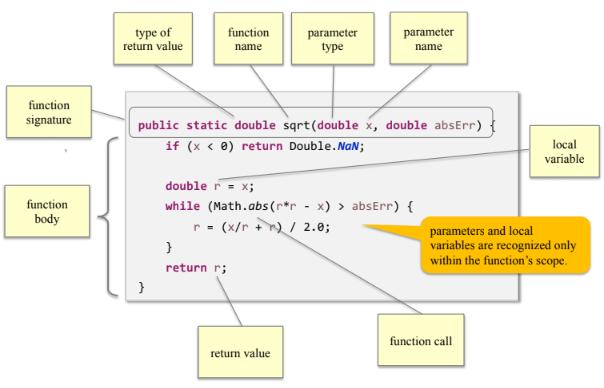
Solution:



* note the end+1 in the answer: s.substring(start,end) is [start,end) = [start, end+1]

**Functions** (10-15 mins)

Let’s recall the function’s structure from the lecture:

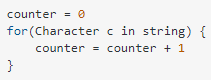


Remind students about writing function:

* not long
* arguments instead of constants
* KISS (Keep It Stupid-Simple)

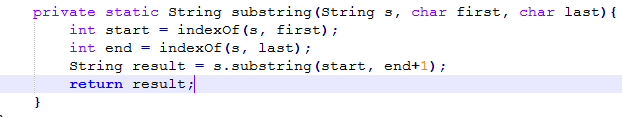
**Warm-up Questions**:

1. Ask students to implement (pseudo-code) s.length()
   1. We want that the students understand that s is an array of chars, and answer is something like:



1. Ask students to implement s.indexOf(char c) as indexOf(String s, char c) **only using s.charAt()!**
   1. Solution can be introduced with bad suggestion that we improve 6 times
      1. indexOf1(), indexOf2(), …indexOf6()
   2. Discuss solution
      1. edge cases
         1. why “int” is the return type and not “long”? In very long string it may fail.
         2. If not exist – what am I returning? It must be different than 0 which is a legitimate answer in this case.
2. Ask students to write a function which receives one String and two characters. The functions should return the substring of the input starting from the first character and terminating at the second character (including both). If one of the characters does not exist or if the last character appears only before the first character - return the empty String.

String substring(String s, char first, char last)



* Note that first,last vars in functionWarmup3 are different from those in function substring! scope is important

Maybe to do a Google Form or a game with questions regarding my notes:

* Vars inside scope that are changing, change the same-named vars outside scope?
  + Loop?, if?, function?
* Whether indexOf1() works well or not? If not, why?
* What are the problems of indexOf2()? (multiple answers are needed here)
* s.indexOf(‘W’) != s.indexOf(‘w’)?
* what is the indexOf(‘ ‘) for s=“asbadasdasd”?
* s1=”123” and s2=”123”:
  + s1.equals(s2) is true
  + s1==s2 is false
  + why?
    - Their name is different
    - That’s a well-known bug in java
    - Strings are compared by reference and not by value
    - You forgot a ‘=’… String compare in java should be by using “===” sign
    - You have an additional ‘=’.. it should be “if(s1=s2)”

**Finding Genes**

For our purposes a DNA strand is considered to be a String in which all characters are one of the four possibilities: ‘A’, ‘C’, ‘G’, ‘T’ (in reality every such letter represents one of four possible biological compounds).

Every three letters compose a single word, or codon (in reality, it codes for amino acids).

Thus, a valid DNA strand should have length divisible by three. Discuss with students how to check that.

A DNA strand can contain many genes. A gene is a sequence of codons inside the strand which is enclosed between two special codos, a start codon and an end codon.

Today, we will only consider one possible start and end codons. The start codon is given by “ATG” and the stop codon is given by “TAG”.

For example if the DNA is "GGTCCGATGCCTGGGCTCTAGGAA", it contains a single gene "ATGCCTGGGCTCTAG".

Another example if given with the DNA "ATGTTCAAATAGCCCTTA" and the gene "ATGTTCAAATAG".

The main task for today is to write a function, which takes a DNA strand as input and returns a single gene which is contained inside it.

Here is the description:

/\*\* Finds the first gene in a given DNA strand, if there is one. Returns the

\* first string of codons (triplets) that starts with "ATG" and ends with "TAG",

\* inclusive. The length of this string must be a multiple of 3.

\* If no such gene is found returns "".

\*/

Before letting the students solve the question, discuss briefly the possible challenges.

* What happens if “TAG” appears before “ATG”.
* What happens if the String looks like this “ATGGTAG”, which is not a valid gene?

Here is the solution:

public String findGene(String dna) {

int start = dna.indexOf("ATG");

if (start == -1) {

return "";

}

int stop = dna.indexOf("TAG", start + 3);

if ((stop - start) % 3 == 0) {

return dna.substring(start, stop + 3);

}

else {

return "";

}

}

**Possible extensions**

Ask students the following questions:

* What can we change if we want to allow for the input String to be with lower case letters as well?
* Suppose we want to find two genes in the DNA strand, instead of one. How can we do it?
* Suppose we want to find ALL genes in the DNA strand, can we even do it, by just using String functions?