Lecture 8

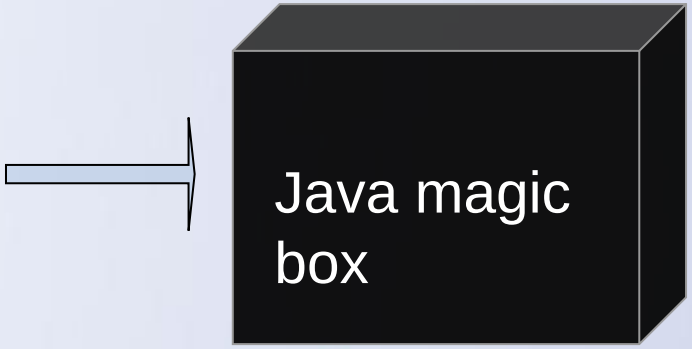
Introduction to Strings
Functions in Java

Before we begin

The scary term “**function**”

- We are going to talk about Java functions today.
- Think of the function as a black-boxed Java magic that does something for us.
- You can put something on the one side of this box and get something different as a result from the other side

Another analogy – a blender! You input fruits and it returns a glass of juice!



IT TALENTS
Training Camp

Introduction to Strings

- Strings are sequences of characters
- Each character is a Unicode symbol
- Strings are immutable (read-only)
- String is a reference type – stored in heap
- Example:

```
String a = "Hi!";
```

Declaration and initialization

- Declaring a String:
  - Not initialized variables have value of null
- Initializing a String:

```
String first; // first is equal to null;

String second = "2nd";
first = "1st";
first = second;
first = second + " some other text";
```

Strings and console

- Reading strings from console
  - Using Scanner method .nextLine();
- Writing strings to console

```
Scanner scanner = new Scanner(System.in);

String st = scanner.nextLine();
//for example st = "My String"
System.out.print(st + "\n");
System.out.println(st);
System.out.printf(st + " %s%s", "is so", " so strong");
```

```
My String
My String
My String
My String is so so strong
```

Comparing Strings

- Comparing strings
  - .equals(…);
  - .equalsIgnoreCase(…);
  - .compareTo(…);
  - .compareToIgnoreCase(…);

```
String a = "Hello";
String b = "hello";
System.out.println(a.equals(b));//false
System.out.println(a.equalsIgnoreCase(b));//true
System.out.println(a.compareTo(b));//-32 - the difference in unicode between the first char that is different
//code of H is 72, code of h is 104. So compareTo returns 72-104 = -32
System.out.println(a.compareToIgnoreCase(b));//0 means they are equal
```

Comparing Strings

- Comparing strings
  - Usage of == and != operators

```
String s1 = "Hello";
String s2 = s1;
System.out.println(s1 == s2);//true

String s3 = "Hi!";
String s4 = "Hi";
String copy = s4+"!";
System.out.println(s3 == copy);//false
```

Always use .equals() when comparing values of Strings

- Interned strings

```
String p1 = "Hello";
String p2 = "Hello";
System.out.println(p1 == p2);//true
```

```
String p3 = new String("Hello");
String p4 = new String("Hello");
System.out.println(p3 == p4);//false
```

Other operations with Strings

- Concatenating strings
  - .concat(…), + and += operators

```
String w1 = "Hello";
String w2 = "Master";
String output = w1.concat(" ").concat(w2);//Hello Master
String output2 = w1 + " " + w2;// Hello Master
```

- St
  - Never concatenate Strings in a loop
  - Use StringBuilder when concatenations needed

Other operations with Strings

- Searching in Strings
  - .indexOf(…);
  - .lastIndexOf(…);
  - .charAt(…)

```
String q1 = "My new Java course is awesome!";
System.out.println(q1.indexOf("Java"));//7- the index of J char
System.out.println(q1.indexOf("IS"));//-1 means "not found"
System.out.println(q1.indexOf("o"));//13 - first occurence of char "o"
System.out.println(q1.lastIndexOf("o"));//26 - last occurence of char "o"
System.out.println(q1.charAt(3));//the char at 3rd index
```

Other operations with Strings

- Replacing in Strings

```
String text = "My wife is a good wife";
String correctText = text.replace("wife", "girl");
System.out.println(correctText);// My girl is a good girl

String otherText = "My phone number is 0878627022 and yours is 0888323321";
String censured = otherText.replaceAll("(08)[0-9]{8}", "***");
System.out.println(censured);//My phone number is *** and yours is ***
```

```
String word = "Table";
String upperWord = word.toUpperCase();//TABLE
String lowerWord = word.toLowerCase();//table
```

Functions and Methods

Declaration

- Return type (boolean, int, String, <any other class>
- Method name (starts with lowerCase, use CamelCase convension)
- Brackets (mandatory)
- List with parameters in the brackets (not mandatory)
- Body – starts with { and ends with }

Functions and Methods

Returned types

- *void* – the method do not return value
- Any other type – object(or primitive) with this type must be returned in the body of method
  - Keyword *return*
- Used for break the execution of the method and return a value
- Void methods can execute return but without value

Example with returned type int

Return type

Method name

Parameters

```
int sum(int a, int b) {
        int sum = a + b;
        return sum;
}
```

Body

Return int value

Introduction to String

Let's write some code!

Tasks

- Write a function to print an array
- Write a function to generate random array of size n
- Write a function to multiply two arrays A and B
- Write a function to find the n-th number of Finonachi sequence
- Write a function to convert Roman Number in Decimal Format
- Write a function to find n! factorial

Recursion

- To iterate is human, to recurse is divine
- Recursion is a function that calls itself
- Recursion may seem like an infinite loop or like a dog chasing its tail – it can never catch it.
- That is true only on certain conditions.

Recursion

- Every recursion should have the following characteristics.
  - A simple **base case** which we have a solution for and a return value.
  - A way of getting our problem closer to the base case. I.e. a way to chop out part of the problem to get a somewhat simpler problem
  - A recursive call which passes the simpler problem back into the method with different parameters (**step** of the recursion)

Recursion

n! factorial

Our base case!

```
public int factorial(int n)
        {
                if (n < 2){
                        return 1;

                }else
                {
                        return n *
factorial(n - 1);
                }
        }
```

Next step

Different parameters

Tasks

- Write a recursive function to find the n-th Fibonachi number
- Write a recursive function to find the Gross Devision Number (Най-голям общ делител)
- for exmple, enter 15 and 10, the result is 5
- Write a recursive function to find the simple fraction
- for example, enter 15/10 and the result should be 3/2

Summary

- Introduction to Strings
- String methods
- What is a function and method
- Methods with returned types not void
- How to use *return* keyword
- Recursion