Java cheat-sheet for CSE 1310 Updated 04/29/2019

Disclaimer: This list is not a comprehensive list for Java. For example the do… while() loop is not included. There may also be topics that are required for the class or covered by other instructors that may be missing. – Alexandra Stefan

import java.util.Scanner;

Scanner in = new Scanner(System.in);

in**.nextInt**(), in**.nextDouble**(),in**.next**(), in**.nextLine**(), (To read a char, read it as a string and process the string.)

in**.hasNextInt**(), in**.hasNextDouble**(), in**.hasNextLine**()

System.out.**println**, System.out.**print**, System.out.**printf**

printf: converters: %d, %f, %s, reserve minimum width: %10.2f, %10s, left align: %-10s, %-10.2f

Integers starting with 0: interpreted as base 8: **023**

Integers starting with 0x: interpreted as base 16: **0x23**

s.**length**(), s.**charAt**(index), s.**toUpperCase**(), s.**toLowerCase**(),

s1.**indexOf**(s2) , s1.**contains**(s2), s.**substring**(start), s.**substring**(start, beforeEnd)

s1.**split**(s2) //s2 can be "," (to split at ,) or "[, .!;]" (to split at either one of: , . ! ;)

s1.**compareTo**(s2), s1.**equals**(s2) (Do NOT use '==' or '<=' for strings)

Math.**floor**(n), Math.**ceil**(n), Math.**round**(n)**,** Math.**abs**(n)**,** Math.**pow**(base,exp), Math.**min**(a,b), Math.**max**(a,b)

Integer.MAX_VALUE, Integer.MIN_VALUE, Double.MAX_VALUE, Double.MIN_VALUE

java.util.Random

Random r = new Random(seed); // for without a seed use: new Random();

double r = r**.nextDouble**(); // returns a double in the range [0, 1)

int n = r**.nextInt**(maxRange); // returns an integer in the range [0, maxRange-1]

(Also: Math.**random**() // cannot choose the seed, returns a double in range [0,1))

String ← int

- **String.valueOf(number)**
- Integer.toString(number)
- " " +number

int ← String

- **Integer.parseInt(myStr)** (Integer <- Integer.valueOf(myStr))

double ← String

- **Double.parseDouble(myStr)** (Double <- Double.valueOf(myStr))

String ← char

- **'A' + ""** // evaluates to "A"

char ← String

- s.**charAt**(i)

System.exit(0); // terminates the **program** (successful termination)

return; return data; // terminates a **method** (can be used in **main**)

break; //stops a loop and exits it

Primitive types: int, double, char, boolean (also: byte, short, long, float)

Classes (that we create objects from): String, Scanner, Random,
Classes (use the class directly): Math, Integer, Double

DeMorgan's laws:
!(a || b) = (!a) && !(b)
!(a&&b) = (!a) || (!b)

	array	ArrayList
Creation	int[] arr2 = new int[10]; // only when declaring arr: int[] arr = {8,1,9,0};	*import java.util.ArrayList;* ArrayList<Integer> arrList = new ArrayList<Integer>(); ArrayList<String> strList = new ArrayList<String>();
Size/length	arr.length	arrList.size()
Read at index i	arr[i]	arrList.get(i)
Modify at index i	arr[i] = newVal;	arrList.set(i,newVal);
Insert element at pos i	Must slide elements	arrList.add(i,newVal); arrList.add(newVal); // adds at the end
Remove element at **index i**	Must slide elements	arrList.remove(i);
Remove element with **value `val`**	Must search for it and slide elements over.	arrList.remove(new Integer(val)); strList.remove("cat");
Import:	import java.util.Arrays;	import java.util.Collections;
String ← Array	Arrays.toString(arr);	arrList + "" // force cast to string because of +
Printing	System.out.print(Arrays.toString(arr));	System.out.print(arrList);
Sorting	Arrays.sort(arr)	Collections.sort(arrList); Collections.sort(arrList,Collections.reverseOrder())
Copy	arr2 = Arrays.copyOf(arr,arr2NewSize); (shallow copy of the items in the array)	strList_2 = new ArrayList<String>(strList); (shallow copy of the items in the list)

Files and Exceptions:

```
import java.util.Scanner;
import java.io.File;
import java.io.PrintWriter;
import java.io.IOException;

// Read from file
Scanner inFile = new Scanner( new File(filenameStr)  );
…. // use as a regular Scanner object
inFile.close();
```

```
// Write to file
PrintWriter out = new PrintWriter("out1.txt");
out.println(…);
out.printf(…);
out.close();

//  to use system dependent newline character write:
out.printf("line 1 in file" + System.getProperty("line.separator")  );
```

```
try {
    // code that may throw an exception
}
catch (Exception e) {
   // action to be taken in case the exception was thrown
}
// For files:   FileNotFoundException
```

```
// Other  useful file and directory manipulations
File f = new File(parentDirName, newDirName);
f.mkdir();  f.mkdirs() ;   // Create directory
f.delete()    //Delete directory
f.list()         // List contents of directory:

f.exists() ;
f.isDirectory() ;  f.isFile()
f.lastModified()
```