**Java syntax**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Print (output) | System.out.println("Hello world"); |  |
| Array | int[] arr1=new int[2]; //first way  int[] arr2=new int[] {2,4,6}; // second way |  |
| Matrix | String[][] mat=new String[3][];  mat[0]=new String[] {"A","B"};  mat[1]=new String[] {"C"};  mat[2]=new String[] {"D","E","F"}; |  |
| ArrayList | **import** java.util.ArrayList;  ArrayList<Animal> arr = **new** ArrayList<>();    arr.add("Volvo"); 🡪 add elements to the ArrayList  arr.get(0); 🡪 access an element in the ArrayList  arr.set(0, "Opel"); 🡪 To modify an element  arr.clear(); 🡪 To remove all the elements in the ArrayList  arr.size();To find out how many elements an ArrayList  Collections.sort(arr); // Sort arr |  |
| random | Math.random()returns a random number between 0.0 (inclusive), and 1.0 (exclusive):  int randomNum = (int)(Math.random() \* 101); // 0 to 100  **int** a =(**int**) (Math.*random*() \* (max(+1) - min)+min);  אם נרצה מספרים בין 5 ל 10 (כולל עשר) 11 5 +5 |  |
|  |  |  |
| Input (user) | import java.util.Scanner; // Import the Scanner class  Scanner input = **new** Scanner (System.***in***); |  |
| **instanceof** | **for** (Store s : arr) {  sum += s.getMonthlyRent();  **if** (s **instanceof** BookStore) //  sumOfBookStore+=1;  } |  |
| Overload  כללי העמסה | ניתן שבמחלקה מסווימת יהיו פונקציות עם אותו השם בתנאי:   * מספר הפרמטרים שהפונקציות מקבלות יהיה שונה(כך הקומפיילר ידע לזהות לאיזה פונקציה לשלוח את הערכים) * או שטיפוס הפרמטרים יהיה שונה |  |
|  |  |  |
| down casting | * **Widening Casting** (automatically) - converting a smaller type to a larger type size byte -> short -> char -> int -> long -> float -> double * **Narrowing Casting** (manually) - converting a larger type to a smaller size type double -> float -> long -> int -> char -> short -> byte   או המרה של סוג המשתנה למשתנה אחר  double myDouble = 9.78;  int myInt = (int) myDouble; 🡪 9 |  |
| down casting  object | המרה של אובייקט למחלקה אחרת (בד"כ ממחלקה בסיסית יותר למחלקה נגזרת)  Fruit fruit = getSomeFruit(); #we dont really know what getSomeFruit is returning so we can check the type of fruit using **instanceof**  **if** (fruit **instanceof** Apple) {  // the object can be casted and the code won't fail  Apple castedApple = (Apple) fruit;  } |  |
|  |  |  |
| While loop | |  | | --- | | int index1=1; | | while(index1<3) { | | System.out.println("While: "+index1); | | index1++; | | } | |  |
| Do while loop | |  | | --- | | int index2=1; | | do { | | System.out.println("Do-While: "+index2); | | index2++; | | }while(index2<3); | |  |
| For loop | |  | | --- | | for(int index3=1;index3<3;index3++) { | | System.out.println("for: "+index3); | | } | |  |
| Empty for | |  | | --- | | for(int index3=1; | | index3<3; | | System.out.println("empty for: "+(index3++))); | | } | |  |
| For eche | for (int e:arr2) { System.out.println(e);  } |  |
| function |  |  |
|  |  |  |