Java Output

System.out.println(anything) // use + to concatenate inside the parentheses  
  
Java input

import java.util.\*;

Scanner scanner = new Scanner(System.in);

int i = scanner.nextInt(); // or nextLong(), nextDouble(), etc

String s = scanner.nextLine(); // reads a whole line, including spaces

ArrayLists

* Creation: List<Integer> list = new ArrayList<Integer>();
* Add things:
  + list.add(x) // inserts x at the end
  + list.add(pos, x) // inserts x at list[pos]
  + list.set(pos, x) // changes list[pos] to x
* Get stuff out
  + list.get(pos) // returns list[pos]
* Other nice functions
  + list.size(), list.contains(x) [returns a boolean], list.indexOf(x), list.remove(pos)

Enhanced for loop

for (int i = 0; i < list.size(); i++) {

System.out.println(list.get(i));

}

for (int x : list) {

System.out.println(x);

}

HashMaps (hashtables, dictionaries, etc). The data structure that associates one thing with another. It allows for O(1) insertion, deletion, and key/value retrieval.

* Creation: Map<String, Integer> map = new HashMap<String, Integer>();
  + creates a map from strings to ints.
* Add things:
  + map.put(key, val) // associates a string key with an int value
* Get stuff out
  + map.get(key) // returns the value the key is associated with
* Other nice functions
  + map.size(), map.containsKey(key), map.keySet() [returns the set of all keys], map.remove(key)

Enhanced for loop over maps:

for (String key : map.keySet()) {

int value = map.get(key); // do something with key and/or value

}

HashSets (a hashtable implementing the Set ADT). This is a data structure that stores a set of items that enables O(1) insertion, deletion, and test for membership.

* Creation: Set<Integer> set = new HashSet<Integer>(); // create a set of integers
* Add things:
  + set.add(x) // adds x to the set
* Test if something is in the set
  + set.contains(x) // returns true or false
* Other nice functions
  + set.size(), set.isEmpty(), set.clear(), set.remove(x)