Python	C++ / STL	Java / the Java Collections
(script, interpretive)	(compiled)	(bytecodes)
`print (3//2)`	`cout << 3/2 << endl;`	`System.out.println (3/2);`

`x, y = map (int, input().split())`	`int x, y;`
	`cin >> x >> y;`
		`import java.util.Scanner;`
		`...`
		`    Scanner input = new Scanner (System.in);`
		`    int x = input.nextInt (), y = input.nextInt ();`

`if ...:`
`    ...`
`elif ...:`
`    ...`
`else:`
`    ...`
	`if (...) {`
	`    ...;`
	`} else if (...) {`
	`    ...;`
	`} else {`
	`    ...;`
	`}`
		(same as C++)

`... if` *cond* `else ...`	*cond* `? ... : ...;`	(same as C++)
`and, or`	`&&, ||`	(same as C++)

`while ...:`
`    ...`
	`while (...) {`
	`    ...;`
	`}`
		(same as C++)

`for ... in ...:`
`    ...`
	`for (auto ...: ...) {`
	`    ...;`
	`}`
		–

`for i in range (n):`
`    ...`
	`for (i = 0; i < n; i++) {`
	`    ...;`
	`}`
		(same as C++)

`for i in range (s, t):`
`    ...`
	`for (i = s; i < t; i++) {`
	`    ...;`
	`}`
		(same as C++)

`for i in range (s, t, inc):`
`    ...`
	`for (i = s; i < t; i += inc) {`
	`    ...;`
	`}`
		(same as C++)

–	`int a [8] = { 0 };`	`int[] a = new int [8]; // 語言規範保證全 0`

–	`int a [] = { 1, 2, 3, 4 };`	`int[] a = { 1, 2, 3, 4 };`
		`// int[] a = new int []{1, 2, 3, 4};`

`x = lambda a : a + 10`	`auto x = [](int a) { return a + 10; };`	–

Python	**C++ / STL**	**Java / the Java Collections**

for c in s:
    ...
	for (auto c: s) {
	    ...;
	}
		for (int i = 0; i < s.length (); i++) {
		    char c = s.charAt (i);
		    ...;
		}

list	vector	ArrayList: List

x = [1, 2, 3] # 中括弧，方括弧	vector <int> x = { 1, 2, 3 };	ArrayList<int> x = newArrayList<int>();
		x.add (1); x.add (2); x.add (3);

m = x [1]	auto m = x [1];	int m = x.get(1);
n = len (x)	auto n = x.size();	int n = x.size();
x.append (4)	x.push_back (4);	x.add (4);
x.pop (1)	x.erase (1);	x.remove (1);

x.remove (3) # removes the 1st occurrence of 3	(use find)	int i = x.indexOf (3);
		if (i != -1)
		    x.remove ();

x.clear()	x.clear(); // x.resize(0);	x.clear();
y = x.copy()	auto y = x; // auto y (x);	ArrayList<int> y = newArrayList<int>(x);

z = x + [4, 5, 6]	vector <int> w = { 4, 5, 6 };
	auto z (x); // auto z = x;
	z.insert (z.end(), w.begin(), w.end());
		ArrayList<int> z = newArrayList<int>(x);
		...
		z.addAll (w);

c = x.count (2)	int c = count (x.begin(), x.end(), 2);
	// or
	auto is2 = [] (int p) { return p == 2; };
	int c = count_if (x.begin(), x.end(), is2);
		Collections.frequency (x, 2);

i = x.index (2)	int i = find (x.begin(), x.end(), 2) - x.begin(); // or find_if	int i = x.indexOf (2);

x.insert (1, 5)	x.insert (x.begin() + 1, 5);
	              // returns x.begin() + 1
		x.add (1, 5);

Python	C++ / STL	Java / the Java Collections
x.reverse()	reverse (x.begin(), x.end());	Collections.reverse (x);
x.sort()	sort (x.begin(), x.end());	Collections.sort (x);
mx = max (x)	int mx = *max_element (x.begin(), x.end())	int mx = Collections.max (x);
mn = min (x)	int mx = *min_element (x.begin(), x.end())	int mn = Collections.min (x);
sm = sum (x)	int sm = accumulate (x.begin(), x.end(), 0)	int sm = MathUtils.sum (x);
x = (1, 2, 3) # 小括弧，圓括弧	tuple, pair	
dict	map	Map

x = { "a": 23, "b": 56, "c": 89 } # key: value
# or 	
x = dict (a=23, b=56, c=89)
# or 	
ks = ('a', 'b', 'c')
x = dict.fromkeys (ks)
x ["a"] = 23
x ["b"] = 56
x ["c"] = 89
	map <string, int> x = {
	    { "a", 23 },
	    { "b", 56 },
	    { "c", 89 }
	};
		Map <String, int> x = new HashMap<>();
		x.put("a", 23);
		x.put("b", 56);
		x.put("c", 89);

v = x ["b"] # k = x.get ("b")	int v = x.find ("b")->second;	int v = x.get ("b");

x ["d"] = 100	x ["d"] = 100;
	// x.insert (make_pair ("c", 100));
		x.put ("d", 100);

Python	C++ / STL	Java / the Java Collections

`for k in x: # for k in x.keys ()`
`    ...`
	`for (auto it = x.begin(); it != x.end(); it++) {`
	`    string k = it->first;`
	`    int    v = it->second;`
	`    ...;`
	`}`
		`for (String k : x.keySet ()) {`
		`   ...;`
		`}`

`for v in x.values():`
`    ...`
`# or`
`for k in x:`
`    v = x [k]`
`    ...`
		`for (int v : x.values ()) {`
		`    ...;`
		`}`

`for k, v in x.items(): # each pair`
`    ...`
		`for (Map.Entry<String, int> e: x.entrySet()) {`
		`    String k = e.getKey ();`
		`    int    v = e.getValue ();`
		`    ...;`
		`}`
		`// or`
		`x.forEach (k, v) -> { ... };`

`n = len (x)`	`int n = x.size();`	`int n = x.size();`

`x ["c"] = 111213`
`       # x.update ({ "d": 1112113 })`
	`x ["c"] = 111213;`	`x ["c"] = 111213;`

`x.pop ("b") # del x [1]`	`x.erase (x.find ("b"));`	`x.remove ("b");`
`x.popitem()`		
`x.clear() # vs. del x`	`x.clear();`	`x.clear();`
`y = x.copy() # y = dict (x)`	`auto y = x; // auto y (x);`	`Map <String, int> y = newHashMap<>(x);`
`v = x.setdefault ("b", 10)`		

`z = { "e": { "f": 3, "g": 5 }, \`
`     "h": { "i": 8, "j": 10 } }`
`# or`
`p = { "f": 3, "g": 5 }`
`q = { "i": 8, "j": 10 }`
`z = { "e": p, "h": q }`
	`map <string, map <string, int>> z = ...`	`...`

Python	C++ / STL	Java / the Java Collections
`set`	`set` (also `set_`... methods on vectors, arrays)	`Set`

`x = { 1, 2, 3 } # x = set (( 1, 2, 3 ))`	`set <int> x = { 1, 2, 3 };`	`Set<int> x = new HashSet<>();`
		`x.add(1); x.add(2); x.add(3);`

`if (3 in x):`
`    ...`
	`if (x.find (3) != x.end()) {`
	`    ...;`
	`}`
		`if (x.contains (3)) {`
		`    ...;`
		`}`

`x.add (4) # x.update ([ 4, 2, 6 ])`	`x.insert (4);`	`x.add (4);`

`x.update ( { 4, 2, 6 } )`	`set <int> w = { 4, 2, 6 };`
	`        // int w [] = { 4, 2, 6 };`
	`x.insert (w.begin(), w.end());`
		`...`
		`x.addAll (w);`

`y = x.union ( { 4, 2, 6 } )`	`set <int> y; y.reserve (x.size() + w.size());`
	`auto it = set_union (x.begin(), x.end(),`
	`w.begin(), w.end(), y.begin());`
	`y.resize (it - y.begin());`
		`Set<int> y = new HashSet<>(x);`
		`y.addAll (w);`

`n = len (x)`	`int n = x.size();`	`int n = x.size();`
`x.remove (3) # vs. x.discard (3)`	`x.erase (x.find (3));`	`x.remove (3);`
`x.pop()`		
`x.clear() # vs. del x`	`x.clear();`	`x.clear();`
`y = x.copy() # y = set (x)`	`auto y = x; // auto y (x);`	`Set<int> y = new HashSet<>(x);`

`z = x.difference (y)`	`set <int> z; z.reserve (x.size());`
	`auto it = \`
	`    set_difference (x.begin(), x.end(),`
	`y.begin(), y.end(), z.begin());`
	`z.resize (it - z.begin());`
		`Set<int> z = new HashSet<>(x);`
		`z.removeAll (y);`

`x.difference_update (y)`		`x.removeAll (y);`

`z = x.intersection (y)`	`set <int> z; z.reserve (x.size() + w.size());`
	`auto it = set_intersection (x.begin(), x.end(),`
	`y.begin(), y.end(), z.begin());`
	`z.resize (it - z.begin());`
		`Set<int> z = new HashSet<>(x);`
		`z.retainAll (y);`

`x.intersection_update (y)`		`x.retainAll (y);`

Python	**C++ / STL**	**Java / the Java Collections**

`z = x.symmetric_difference (y)`	`set <int> z; z.reserve (x.size() + w.size());`
	`auto it = set_symmetric_difference \`
	`(x.begin(), x.end(),`
	`y.begin(), y.end(), z.begin());`
	`z.resize (it - z.begin());`
		(union - intersection)

`x.symmetric_difference_update (y)`		(union - intersection)

`if (x.isdisjoint (y))`
`...`		

`if (x.issubset (y))`
`...`
	`if (includes (y.begin(), y.end(),`
	`x.begin(), x.end()) {`
	`...;`
	`}`
		`if (x.contains (y) {`
		`....;`
		`}`

`if (x.issuperset (y))`
`...`
	`if (includes (x.begin(), x.end(),`
	`y.begin(), y.end()) {`
	`...;`
	`}`
		`if (y.contains (x) {`
		`....;`
		`}`

	`deque`	`Deque`
	`list`	
	`priroity_queue` (`top()` at right)	
	`stack`	`Stack: List`
	`queue`	`Queue`