Java from Csharp - 10/31/2014

TopCoder - KawigiEdit - Configuration: General/Testing  
C:\Users\owner\Documents\Visual Studio 2008\Projects\TopCoderCompetition - for C#  
**C:\Users\owner\My Programs\JavaEclipse\TopCoder\src** - for Java  
Change language from C# to Java by choosing regular editor, then Display Preferences, then Editors, Java

Differences with C#

**C# Java**   
aArray.Length aArray.length   
sStr.Length sStr.length()   
stArray[1].Length stArray[1].length()

st.Replace(“-“,””).Length st.replaceAll(“-“,””).length()  
String.join("", stArray) Java 8 has join() as two class methods on the String class  
Math.Abs() Math.abs()  
Math.Min() Math.min()  
Math.Ceiling() Math.ceil()  
int.MaxValue; Integer.MAX\_VALUE (int)1e9;  
d = Convert.ToDouble(st) d = Double.parseDouble(st)  
**st.Substring(ind, len) st.substring(begInd, endInd)**  // (7,6) becomes (7,13), **less than endInd!**

**... ind can be greater than length of string and not cause abend.**  
s[i] s.charAt(i)  
int[,] a int[][] a  
string s String s  
s.Contains() s.contains()  
if (st.Contains('c')) if (st.contains("c")) //need to find a string with double quotes  
if (st.Contains(st2[i])) if (st.contains(st2.substring(i,i+1))) //find string of 1 char at pos i  
String.Compare(s1,s2) >= 0 s1.compareTo(s2) >= 0  
Array.Copy(a, b, k); System.arraycopy(a,0,b,0,k);  
Array.Copy(a, k, c, 0, n - k); System.arraycopy(a,k,c,0,n-k);  
**string[] w = new string[9] String[] w = new String[9] //initializes to null instead of ""  
 Arrays.fill(w, "") //initialize string array to empty string ""  
if (str == "") if (str.equals(""))   
if (st == "UNION") if (st.equals("UNION")) //former worked on my machine but not in TC!  
if (st == currentSt) if (st.equals(currentSt))**  
s[i][j].ToString() String.valueOf(s[i].charAt(j)) //s[i] is array of Strings  
**num.ToString()**  **Integer.toString(num)** or **String.valueOf(num)** or **(num+"")** !!!’

Convert.ToInt32(st); Integer.valueOf(st); or Integer.parseInt(st);  
c.ToString().ToUpper() Character.toString(c).toUpperCase(); or (c+"").toUpperCase(); or  
 String.valueOf(Character.toUpperCase(c)) or **Character.toUpperCase(c)**  
Convert word to upper chars char[] c = word.toUpperCase().toCharArray();  
c = char.ToUpper(c); c = Character.toUpperCase(c);   
Convert char[] c to string String s = **new String(c)**

Convert char[,] to string[] char[][] b = new char[n][n]; **s[i] = new String(b[i])**  
Array.Sort(a) Arrays.sort(a) //also import java.util.\* or java.util.Arrays  
Array.Reverse(a) Collections.reverse(a) //List<Integer> a = new ArrayList<Integer>();

Arrays.sort(a, Collections.reverseOrder()); // **Integer[] a** (not primitive int[] a)  
List<int> d..; **d.sort();** ArrayList<Integer> d...; **Collections.sort(d);**List<int> d..; d.Remove(i) ArrayList<Integer> d...; d.Remove((Object)i); //object i vs. index i  
List<int>[] t = new List<int>[3]; ArrayList<Integer>[] t = new ArrayList[3]; //**Dropped <Integer>**  
... for (int i = 0; i < 3; i++) ... for (int i = 0; i < 3; i++)

... t[i] = new List<int>(); ... t[i] = new ArrayList<Integer>();

List<long> col = new ArrayList<Long> col = new ArrayList<Long>(Arrays.asList(r, g, b ));

... List<long>(new long[] { r, g, b })

Array.Copy(students, st2, n); st2 = Arrays.copyOf(students, n); //students, st are int[]  
foreach (char c in st) for (char c : st.toCharArray()) //char[] charArray = st.toCharArray();  
Dictionary<int,bool> d = new HashMap<Integer, Boolean> d = new HashMap<Integer,Boolean>();  
 ... Dictionary<int,bool>(); HashSet<Integer> set = new HashSet<Integer>(); d.add(i);  
... d[num] = true **d.put(num, true)** //put, get(num) - **HashMap**  
... d.Count d.size()  
n/a if (!Character.isAlphabetic(ch))

List<string> s = new List<string> ArrayList<String> s = new ArrayList<String>() //import java.util.\*  
List<int[]> p = new List<int[]>() ArrayList<int[]> p = new ArrayList<int[]>()  
List<int> p = new List<int>() ArrayList<Integer> p = new ArrayList<Integer>()  
List<type> ArrayList<type> .size(), .get(), .set(), .add(), .clear(), .contains()  
listP[5] listP.get(5);  
List<char> ArrayList<Character> or TreeSet<Character>  
... TreeSet<Integer> ts = TreeSet<I...>(); ts.add(j); ts.remove(i); **ts.ceiling(i);**  
return ans.ToArray() int[] ret = new int[ans.size()];  
: for (int i = 0; i < ans.size(); i++) ret[i] = ans.get(i);  
: return ret; //ArrayList<Integer> ans -- need to return int[]  
: or **ans.toArray(new String[0]); ans.toArray(new Int[0]); ??**p.Count p.size()  
p.Add(num) p.add(num)  
p[i][j] p.get(i)[j]  
aArray.CopyTo(a2Array, 0) System.arraycopy(aArray, 0, a2Array, 0, aArray.length)  
string.Format("${0:#,###,###,##0.00}", a) String.format("$%,d.%02d", dollars, cents);  
 DecimalFormat fmt = new DecimalFormat(("'$'#,###,###,##0.00");  
 fmt.format(dollars + (cents/100.00)) //needs java.text.\*  
9 decimal places System.out.format("%.9f%n", ans);  
Console.WriteLine(min.ToString  
("F3", CultureInfo.InvariantCulture)); out.format("%.3f", min); //3 decimals  
Console.WriteLine(intNum / (double)num); out.println((int)intNum / (double)num); // x.0 -> x only

string s = Console.ReadLine(); BufferedReader in = new BufferedReader(new

InputStreamReader(System.in));

String s = in.readLine(); //append to main: throws Exception  
 Scanner in = new Scanner(System.in);

String s = in.nextLine();  
Console.ReadLine() sc.next().split(" ") //Scanner sc = new Scanner(System.in);  
 sc.nextLine() //Use extra after sc.nextInt() before real sc.nextLine()

string[] s = Console.ReadLine().Split(); ??

string [] s = st.Split(':') String [] s = st.split(":"); //Note: double quotes instead of single quotes  
string[] s = st.Split('+', '='); String[] s = st.split("[+=]"); //uses regex within double quotes  
string[] s = st.Split('+'); String[] s = st.split("[+]"); //+ can't be used without brackets, in regex

int L = int.Parse(s[0]); int L = sc.nextInt();

int R = int.Parse(s[1]); int R = sc.nextInt();

S.Insert(i, S) String st = S.substring(0,i) + S + S.substring(i);

int.Parse(s) Integer.parseInt(s) or Double.parseDouble(s)

sb.ToString() sb.toString() //but System.out.println(sb) will convert sb to String  
sb.Append(c) sb.append(c)  
sb.Append(num.ToString()) sb.append(num) or sb.append(num + "") or String.valueOf(num)  
sb.AppendLine(); sb.append("\r\n");  
sb.Remove(sb.Length - 1, 1); sb.deleteCharAt(sb.length() - 1);  
sb.Remove(sb.Length - 1, 3); sb.delete(sb.length() - 1, sb.length() - 1 + 3); //start index, end index  
sb[index] sb.charAt(index)  
public static void Main() public static void main()  
bool b boolean b  
const c = int.MaxValue; final c = Integer.MAX\_VALUE //constants all upper, separated by \_  
c = Convert.ToChar(c - 5); c = (char)(c - 5);

new Random(Environment.TickCount); new Random();  
ra = rnd.Next(0, i); ra = rnd.nextInt(i+1);

ra = rnd.Next(0, 50); Random rand = new Random(); int value = rand.nextInt(51); //0 - 50

int random = (int )(Math.random() \* 50 + 1); //0 - 50

int[] d = new int[3] {240, 12, 1}; int[] d = new int[] {240, 12, 1}; //remove array dimension of 3

st = String('.', n) String st = ""; for (int i=0; i<n; i++) st+=".";

//C#: Override usual compare method if one exists,

//or establish the compare method for an object without one.

Array.Sort<Player>(player, new Comparison<Player>(Compare));

static public int Compare(Player a, Player b)

{

//Sort according to highest percent then height (total)

if (a.percent > b.percent) return -1;

if (a.percent < b.percent) return 1;

if (a.percent == b.percent)

if (a.height > b.height) return -1; else return 1;

return 0;

}

//Java  
Arrays.sort(player);

**class** Player **implements** Comparable<Player>

{

**...**

**public** Player(String nam, **int** percent, **int** height)

{

**...**

}

//Comparable interface: sort using a type's natural order.

//The type Player must implement the inherited abstract method

//Comparable<Player>.compareTo(Player)

**public** **int** compareTo(Player that)

{

**if** (**this**.percent > that.percent) **return** -1;

**if** (**this**.percent < that.percent) **return** 1;

**if** (**this**.percent == that.percent)

**if** (**this**.height > that.height) **return** -1; **else** **return** 1;

**return** 0;

}

}

Comparable interface: sort using a type's natural order.

import java.io.BufferedReader;  
import java.io.InputStreamReader;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));   
int number=Integer.parseInt(br.readLine());

TopCoder automatically adds 4 imports:

**import** java.util.\*;

**import** java.util.regex.\*;

**import** java.text.\*;

**import** java.math.\*;

import java.awt.geom.\*; //I've seen some add this one

Wata uses the following:  
import java.util.\*;  
import static java.lang.Math.\*;  
import static java.util.Arrays.\*;

import java.util.\*; //for ArrayList<>, Arrays.sort

For some reason, CodeChef does not like "public" preceding the class and needs to throw exception:

**~~public~~ class** CC2013\_10\_B\_PPNUM {

**public** **static** **void** main(String[] args) **throws** IOException {

Scanner sc = **new** Scanner(System.*in*);

System.*out*.println("Working Directory = " + System.*getProperty*("user.dir"));