The Sets class

/\*\* A library of operations on sets of integer values. A set is a collection of values without

\* duplicates and without order. For example, 2, 5, 2, 7 is not a set. {2, 5, 7} is a set.

\* Because the order is insignificant, {2, 5, 7} is the same set as {5, 7, 2}. \*/

public class Sets {

public static void main(String[] args) {

int[] s1 = { 9, 8, 12, 7 };

int[] s2 = { 6, 12, 7, 9, 8 };

int[] s3 = { 7, 8, 4 };

int[] s4 = { 12, 9 };

println(intersection(s1, s2, s3));

System.out.println(dominates(s4, s3));

println(oddValuesOf(s2));

}

/\*\* Checks if the value e appears in the set.

\* For example, if the set is { 7, 2, 5 } and e is 2, returns true.

\* if the set is { 7, 2, 5 } and e is 3, returns false. \*/

public static boolean elementOf(int e, int[] set) { // Code not shown }

/\*\* Returns the intersection of the two sets. The intersection of set1 and set2

\* is the set containing the elements that are both in set1 and in set2. \*/

public static int[] intersection (int[] set1, int[] set2) { // Code not shown }

/\*\* Returns the intersection of the three sets. The intersection of set1, set2, and set3

\* is the set containing the elements that are both in set1, set2, and set3. \*/

public static int[] intersection (int[] set1, int[] set2, int[] set3) {

// Question 2

}

/\*\* Checks if set1 dominates set2. Set1 dominates set2 if every element of set1

\* is strictly greater (>) than every element of set2. \*/

public static boolean dominates(int[] set1, int[] set2) {

// Question 4

/\*\* Returns a set containing the odd elements of the given set (elements that are not multiples of 2). \*/

public static int[] oddValuesOf(int[] set) {

// Question 5

}

/\*\* Prints the elements of the set, and then moves the cursor to the next line. \*/

public static void println(int[] set) { // Code not shown }

}

**Executing the program**

% java Sets

{ 8, 7 }

true

{ 7, 9 }

The Tables class

public class Tables {

public static void main(String args[]) {

int[][] a = { { 1, 2, 4, 0 },

{ 0, 1, 0, 2 },

{ 2, 3, 5, 1 } };

println(colAverages(a));

}

/\*\* Returns a one-dimensional array containing the averages of the columns of the given

\* table (a two-dimensional array in which all rows have the same number of elements). \*/

public static double[] colAverages(int[][] arr) {

// Question 6

}

/\*\* Prints the given one-dimensional array, and moves the cursor to the next line. \*/

public static void println(double[] arr) { // Code not shown }

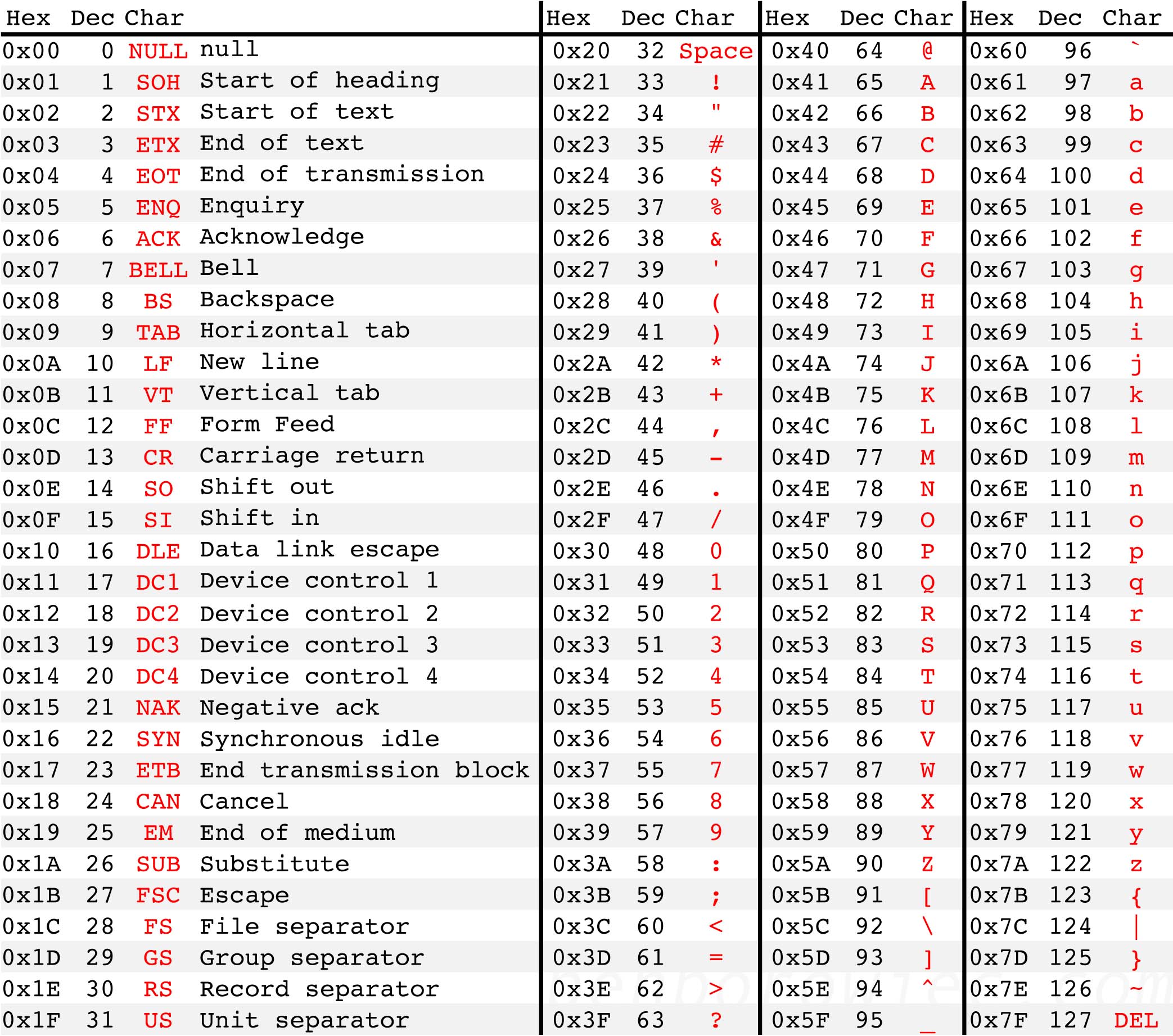
**Executing the program**

% java Tables

1.0 2.0 3.0 1.0

ASCII codes

The following table lists the character codes of a subset of commonly used characters:



Arithmetic operations

In case you will need this operation:

The “modulo” operation x % y computes the remainder (שארית) of the integer division x / y.

For example, 17 % 5 gives 2.