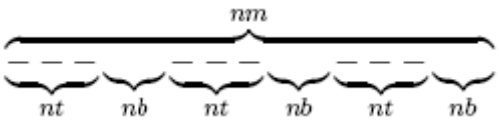
LABs: Loops, Arrays, and Methods

LAB1

Write a method with the signature public static void drawLine(int nm, int nt, int nb) that displays a dashed line composed of hyphens and spaces to the console, as shown in the following figure.



The line is composed of **nt** hyphens (-) followed by **nb** spaces. This pattern should repeat **nm** times. The pattern shown in the figure above was obtained with the call :
drawLine(3, 3, 2);.

LAB2

Write a method with the signature public static boolean isPalindrome(String s) that takes a String object as a parameter and tests whether it is a palindrome. A palindrome is a phrase that reads the same forwards and backwards, for example, level, *radar*, or "Was it a car or a cat I saw".
You can either transform the string into a character array using the toCharArray method or directly use the charAt and length methods of the String class. Also, remember to handle spaces, which should be ignored.

LAB3

Write a sum method that takes a one-dimensional integer array int as a parameter and returns the sum of the integers contained in the array. Here is the method signature: public static int sum(int[] tab);
For example, the call sum(new int[] {1, 2, 3, 4, 5}); returns 15.

Lab4

Write a method with the signature: public static int bintodec(String bin) that takes a string composed only of 0s and 1s representing a binary number as a parameter and converts it to a decimal number.
For example, the call bintodec("001001") returns 9 (2^3 + 2^0).

LAB5

Write a min method that takes a one-dimensional array of double floating-point numbers as a parameter and returns the smallest element in the array (the one with the smallest value). Here is the method signature:
public static double min(double[] tab);
For example, the call min(new double[] {-9.5, 1, -98.21, 218.2}); returns -98.21.

Lab6

Write a compareTab method that takes two one-dimensional integer arrays int (tab1 and tab2) of the same length as parameters. The method should return a character array (char) of the same length containing the characters <, =, and >. The i-th element of the returned array contains < if tab1[i] < tab2[i] (same for = and >). Here is the method signature:
public static char[] compareTab(int[] tab1, int[] tab2);
For example, the call compareTab(new int[] {1, 2, 3}, new int[] {3, 2, 1}); returns the array ['<', '=', '>'].

Some methods related to strings and the Math class

1. The toLowerCase() method transforms any alphabetic character into its lowercase equivalent.

```
String chaine = new String("COUCOU TOUT LE MONDE !"), chaine2 = new String();
chaine2 = chaine.toLowerCase(); // Gives "coucou tout le monde !"

String chaine = new String("COUCOU TOUT LE MONDE !"), chaine2 = new String();
chaine2 = chaine.toLowerCase();   //Donne "coucou tout le monde !"
```

2. The toUpperCase() method is the opposite of the previous one.
3. The length() method returns the length of a string (including spaces).
4. The equals() method allows you to check (and test) if two strings are identical. This is the function you will use for conditional tests on String.
5. The result of the charAt() method is a character: it is a method for extracting a character. It can only operate on String.
6. The substring() method extracts a part of a string. It takes two integers as arguments: the first defines the first character (included) of the substring to extract, and the second corresponds to the last character (excluded) to extract. Again, the first character is at index 0.
7. The indexOf() method searches a string for a given sequence of characters and returns the position (or index) of the substring passed as an argument. The indexOf() method searches from the beginning of the string, while lastIndexOf() searches from the end but returns the index from the beginning of the string. Both methods take a character or a string as an argument and return an int. Like charAt() and substring(), the first character is at index 0. Here is an example:

```
String mot = new String("anticonstitutionnellement");
int n = 0;
n = mot.indexOf('t'); // n is 2
n = mot.lastIndexOf('t'); // n is 24
n = mot.indexOf("ti"); // n is 2
n = mot.lastIndexOf("ti"); // n is 12
n = mot.indexOf('x'); // n is -1
```

8. The methods we will see require the Math class, which is part of java.lang. It is therefore one of the foundations of the language.

```
double X = 0.0;
X = Math.random(); // Returns a random number between 0 and 1, like 0.0001385746329371058
double sin = Math.sin(120); // The sine function
double cos = Math.cos(120); // The cosine function
double tan = Math.tan(120); // The tangent function
double abs = Math.abs(-120.25); // The absolute value function (returns the number without the sign)
double d = 2;
double exp = Math.pow(d, 2); // The exponent function
// Here, we initialize the variable exp with the value of d raised to the power of 2
// The pow() method takes a value as the first parameter and an exponent as the second
```

9. Another way to iterate through an array:

```
public class Sdz1 {
   public static void main(String[] args) {
      String[] a = {"toto", "tata", "titi", "tete"};
      Through_array (a);    }

   static void Through_array (String[] tabBis) {
      for (String str : tabBis) {     System.out.println(str);       }    }}
```