Lab session 1: Programming Fundamentals

Note: In this first lab session you are only allowed to make use of constructs that are taught in the first week of the course (so no loops, recursion, arrays, functions, etc).

Problem 1: Hello world!

The first exercise is to make a small computer program that outputs `Hello world!` on the screen. The goal of the exercise is to get acquainted with the programming environment on the lab computers and the automatic assessment system *Themis*.

1. Log on to the lab system. After log in, open a terminal window. In this terminal you can type commands. Start by creating a directory `progfun` that you will use for the course *Programming Fundamentals*. You create this directory with the command:

   ```
   mkdir progfun
   ```

2. Navigate to this directory using the command `cd` (change directory).

   ```
   cd progfun
   ```

3. Create another directory `lab1`, which is a subdirectory of `progfun`. Create in the directory `lab1` another subdirectory `hello` and navigate to this directory:

   ```
   mkdir lab1
   cd lab1
   mkdir hello
   cd hello
   ```

   Create the file `hello.c` using an editor (for example `geany`).

   ```
   geany hello.c &
   ```

   Type the following program, replace `...` by the right information, and save it.

   ```
   /* file:    hello.c              */
   /* author:  ...... (email: ....) */
   /* date:    ......               */
   /* version: .....                */
   /* Description: This program prints 'Hello world' */

   #include <stdio.h>
   #include <stdlib.h>
   #include <math.h>

   int main(int argc, char *argv[]) {
     printf("Hello world!\n");
     return 0;
   }
   ```

4. Compile the program using the command:

   ```
   gcc -std=c99 -Wall -pedantic hello.c
   ```

5. On successful compilation, an executable file `a.out` is produced. Run the program:

   ```
   ./a.out
   ```

6. If you are convinced that your program works well, you can submit it to the online assessment system Themis. You can reach Themis via the link `https://themis.housing.rug.nl` using your favourite webbrowser. You have completed this first task once Themis accepted your submission.

Problem 2: Camping

The Jones family has set up the tent on a camping site. The weather is hot and for the kids they want to fill a pool with water. Father has two jerrycans, each having a volume of 12 litres. He is strong enough to walk with the jerrycans to the nearest water tap and carry them back completely filled.

Write a program that asks how many litres of water is needed to fill the pool. You may assume that this number is an integer. The program must print how often dad must walk up and down to fill the bath.

Example 1:
input:
```
240
```
output:
```
10
```

Example 2:
input:
```
480
```
output:
```
20
```

Example 3:
input:
```
2400
```
output:
```
100
```

Problem 3: Arithmetic expression

The input for this exercise consists of a line that has the form `aXbYc`, where `a`, `b`, and `c` are non-negative integers, while `X` and `Y` are chosen from `+` and `*` (see example inputs below). The output of your program must be the value that results from evaluating the expression on the input. Note that the input contains no spaces or tabs.

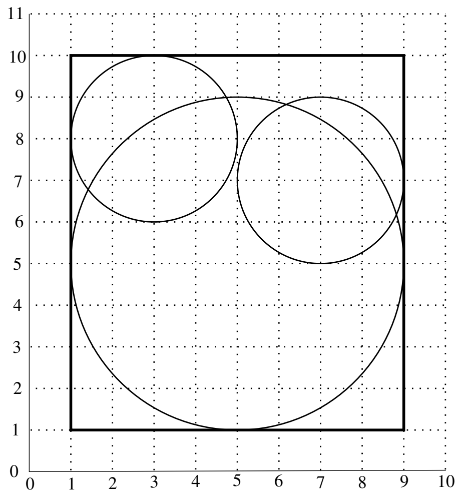
Example 1:
input:
```
2+3+5
```
output:
```
10
```

Example 2:
input:
```
2*3*5
```
output:
```
30
```

Example 3:
input:
```
2*3+5
```
output:
```
11
```

Problem 4: Bounding Box

In the figure on the right, you see a grid containing three circles. The largest circle is defined by the center coordinate $(5, 5)$ and has radius 4. The smaller circles both have radius 2. The left one has its center at $(3, 8)$, while the right one has its center at $(7, 7)$. The rectangle is the smallest axes-aligned rectangle that surrounds the three circles completely. This rectangle is called the *bounding box* of the three circles. The bottom left corner point of this bounding box is the grid point $(1, 1)$, and the top right corner is the point $(9, 10)$.



Make a program that reads from the input the center points and radii of three circles. Per line there are three integers. The first two are the coordinates of the center point (x, y) of a circle. The third is its radius. The program must print on the screen the bottom left corner point and the top right corner point of the bounding box of the three circles. Make sure that your program produces output in the exact same format as given in the following examples.

Example 1 (see figure):

input:
```
5 5 4
3 8 2
7 7 2
```
output:
```
[(1,1),(9,10)]
```

Example 2:

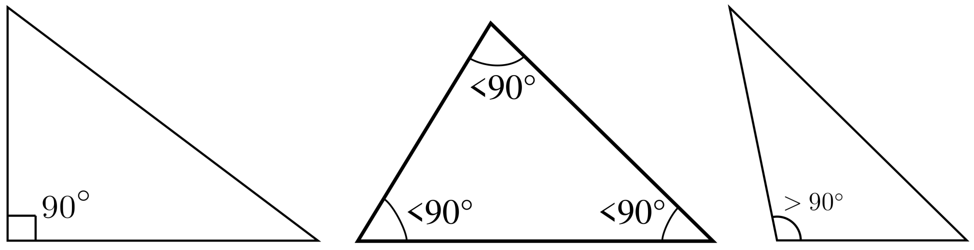
input:
```
2 3 2
5 4 2
4 7 1
```
output:
```
[(0,1),(7,8)]
```

Example 3:

input:
```
5 5 5
4 4 4
3 3 3
```
output:
```
[(0,0),(10,10)]
```

Problem 5: Triangles

Recall that a *rectangular triangle* is a triangle in which two sides are perpendicular. An *acute triangle* is a triangle with three acute angles (less than 90 degrees), while an *obtuse* triangle has one obtuse angle (greater than 90 degrees).



rectangular triangle   acute triangle   obtuse triangle

Make a program that reads from the input three positive integers a, b, and c. Each of these numbers is at most 1000. These numbers represent the lengths of the sides of a (possible) triangle. The output of your program must be `RECTANGULAR` if a rectangular triangle can be formed with these lengths. If an acute triangle can be formed, then the output must be `ACUTE`, and the output must be `OBTUSE` in case an obtuse triangle can be formed. The output must be `NONE` if none of these cases apply.

Example 1:

input:

```
3 4 5
```

output:

```
RECTANGULAR
```

Example 2:

input:

```
3 4 6
```

output:

```
OBTUSE
```

Example 3:

input:

```
3 4 4
```

output:

```
ACUTE
```