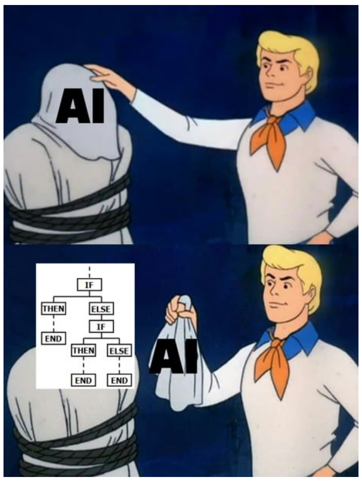
Computer Architecture

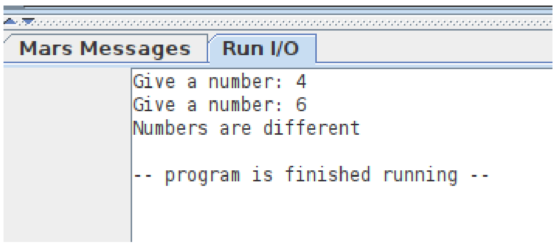
Exercise 4
(if-then-goto)

Peter Stallinga
Universidade do Algarve 2024-2025



If-then-goto in assembler jargon is called branching and we have instructions like `beq` (branch if equal), `bne` (branch if not equal) and `bge` (branch if equal or greater than).

If-then: Write a MIPS program that asks two integer numbers and compares them



```
Give a number: 4
Give a number: 6
Numbers are different

-- program is finished running --
```

If-else: Write a MIPS program that asks two integer numbers and determines which one is bigger, or if they are equal.

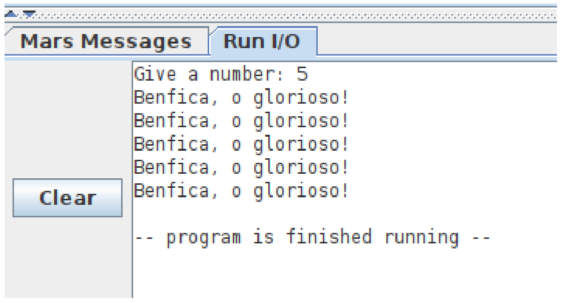
Remember, **in MIPS only the concept of if-then-goto exists**. Loops (for, while, do-while) do not exist. We have to implement that ourselves. For instance for-loops and do-while loops:

Remember, in C it is written as

```
  for(i=0; i<10; i++)
    printf("%s", hellow);
```

Now, imagine that the iteration variable i is stored in register `$t0`, the end value 10 is stored in `$t1`, and we have the branching instruction "if (condition) then goto". (See the MIPS Reference Card for conditions used in branching). How to do the following?

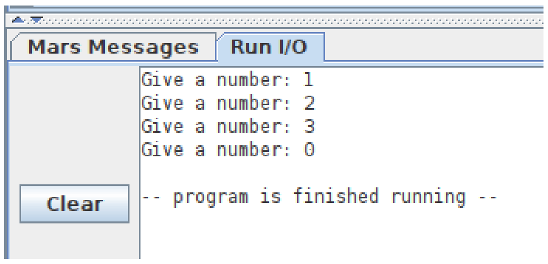
For-loop: Write a MIPS program that asks for a number *n* and prints *n* times the text "Benfica, o glorioso!"



```
Give a number: 5
Benfica, o glorioso!
Benfica, o glorioso!
Benfica, o glorioso!
Benfica, o glorioso!
Benfica, o glorioso!

-- program is finished running --
```

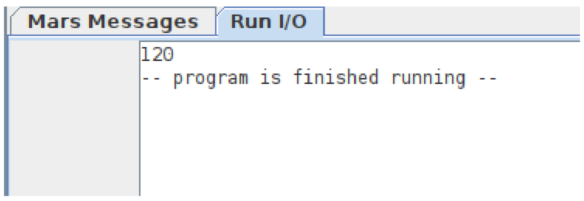
Do-while: Write a MIPS program that asks numbers until the number is 0.



```
Give a number: 1
Give a number: 2
Give a number: 3
Give a number: 0

-- program is finished running --
```

Write a MIPS program that calculates the factorial of a number, *n*! Example for 5!:



```
120
-- program is finished running --
```

The relevant instructions for today are:

`j`	(Unconditional) jump to address
`beq`, `bne`, `bge`	Conditional jump ('branch') to address
`move`	Move (copy!)
`addi`	Add immediate
`li`	Load immediate
`la`	Load address