# Interpreter for course - Analysis of Programing Languages

Martin Orem, Matúš Kyseľ, Lukáš Turčan

# Pseudo-code

The idea of this language is based on simple fact that almost all algorithms are already written in pseudo code.

So why not write a simple interpreter for that?

## Build

To build this interpreter just run bash script `./make.bash`

### Requirements

For building is necessary flex and bison and gcc

## PseudoCode

### Basic Operation

Like every language even ours PseudoCode supports basic arithmetic operation as `+-/*<>!=`. Each vraiable must be defined this way `A`, but currently we are supporting only integer. Arrays can be defined similary `A = 1,2,3`

### Loops

This language support only one type of loops. It's basic `for` loops with syntax like that

```
 for i from 1 to N do     print i  end for
```

### Conditions

This language support just basic `if` and `else` condtion with syntax like that

```
if A > B then           //DO SOMETHING end if
```

or example of if-else statement

```
if A > B then           //DO SOMETHING else     //DO SOMETHING ELSE end if
```

### Functions

Functions are defined just with special keyword `func` and every function must be ended with function ending `end func`. Here is simple example

```
func foo( A )     foo = A + 1 end func
```

## Examples

As for every language let's start with the most important

```
print 'Hello World!'
```

Next example is simple bubble sort on array of integers.

```
func bubblesort( a )
  N = len(a)
  for i from 1 to N do
        for j from 0 to N - 1 do
        if a[j] > a[j + 1] then
          swap( a[j], a[j + 1] )
        end if
        end for
  end for
end func
```