Belinda (*.bl)

- **Creators:**

Thomas Bremond

Benjamin Dran

Valentin Nelu Ifrim

- **Types:**

Ineger 32bit: I

If you need to check for condition 0 is false, and everything else is true. The default value for true is 1

Char 8bit: C

A character is surrounded by **|** to obtain its acii value

```
var <- 72;

var <- |H|;
```

being *var* a declared C variable.

To identify string we use arrays of characters and you can assign an array of characters using surrounding it with **||** .

```
string <- ||Hello World!||, |\0| ;
```

(see array assignment later)

The **|\0|** is the end string character (as in C).

Constant : CONS

It is an integer value that doesn't change during the execution of the program.

- **Assignment statement:**

We use **->** o **<-** to assign a value to a variable: the arrow point to the destination of the value.

```
5 -> var;

var <- 5;
```

being var an **I** (integer 32bit) variable.

The assignment can be done only on the **do** block except for the constant which are done in the **declare** block like in the following example:

```
CONS var 5;
```

that creates a constant with the value of 5.

• Operators

Integer Operators			
Name	**Symbol**	**Example**	**Result (X value)**
Addition	.+	X <- 5 .+ 7	12
Subtraction	.-	X <- 9 .- 7	7
Multiplication	.*	X <- 6 .* 3	18
Division	./	X <- 15 ./ 7	2
Modulo	.%	X <- 15 .% 7	1
Boolean operators			
And	.and	X <- 1 .and 0	0
Or	.or	X <- 1 .or 0	1
Negation	.not	X <- .not 0	1
Greater	.>	X <- 15 .> 7	1
Lower	.<	X <- 12 .< 6	0
Equal	.=	X <- 15 .= 15	1
Greater or equal	.>=	X <- 11 .>= 8	1
Lower or equal	.<=	X <- 11 .<= 7	0

• Structured data type

Arrays of integers or characters. To declare an array you have to use square brackets after the data type.

```
I[] array;
```

Declaration of an array of integers.

By default if there is not any number in the square brackets the dimension of the array is 100. If you want a different dimension you have to specify it in the brackets;

```
I[20] arrayOf20elements;
```

Declaration of an array of 20 integers.

If you want to assign the values of an array you can assign them one by one or all together. That has to be done in the **do** block.

```
vet[0] <- 5;
```

Assign the value *5* to the first element of the array *vet*.

```
vet <- 4, 26, 7, 25, 69;
```

Assign the values *4, 26, 7, 25, 69* to the first 5 elements of the array *vet*.

By default the value assigned to each element of the array is **0** for integers and **\0** for characters.

• Statements for selection and repetitions

Belinda language has **if** statements and **switch** for making decisions and **for** and **while** for loops as shown in the examples.

If statement

```
if (boolean expression) start
          ___code;
else if (boolean expression) start
          ___code;
else start
          ___code;
fi
```

Switch (part 1)

```
switch (variable):
          case value1 start
          case value2 start
                    __code;
                    end
          case value3 start
                    __code;
                    end
```

Switch (part 2)

```
          default start
                    code__;
                    end
switchend
```

While loop

```
while (boolean expression) start
        __code;
whileend
```

For loop

```
for(var assignment; boolean expression; operation) start
        __code;
Forend
```

Subroutines

There can be subroutines which can be invoked in the **do** block. The syntax is:

```
[Return type] funcName(I par1, I par2) start
        __code;
        return var;
funcend
```

Input and output

For input and output it is used the **Console** object and then invoke on it the methods *print* and *scan* as shown in the example.

Console..print

```
C var;
…
var <- |H|;
Console..print(var);
Console..print(|S|) ;
```

Console..scan

```
C var;
…
var <- Console..scan();
```

Further information

To comment a single line use **--,** to comment multiple lines surround the text with **!-- text comment --!**.

We use the **end** keyword to exit loops or switches like the **break** keyword in C.

To end each instruction we use the semicolon **;**.