

# Lesson 2 - Introduction to C language

## Logical Computational Thinking

**Stefano MARTINA**

stefano.martina@gmail.com



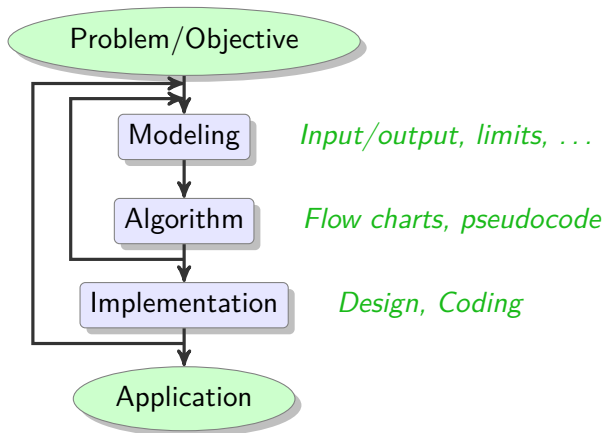
Scuola Leonardo Da Vinci (Firenze)

10 September 2015

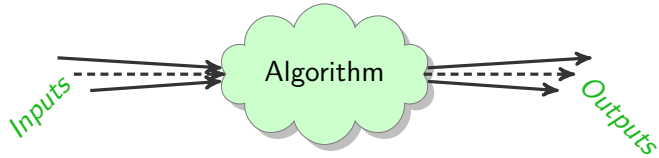


This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).

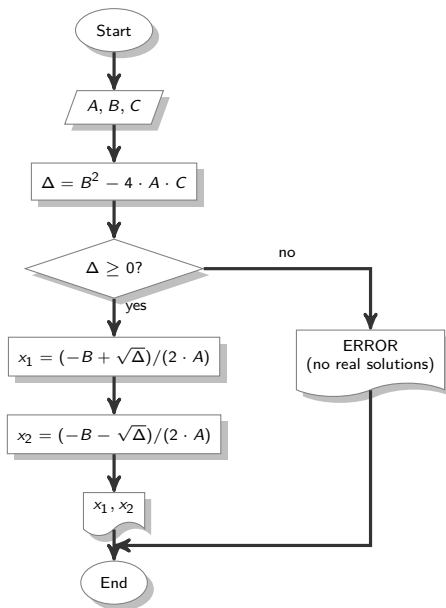
# Review



# Model



# Algorithm



# C language

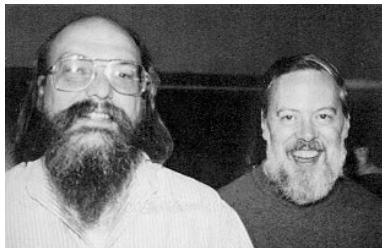


Figure: Ken Thompson and Dennis Ritchie

## History

- ✓ The language **C** is developed in the 70's by Dennis Ritchie
- ✓ Along with the **Unix** system, created by Ken Thompson and Dennis Ritchie in the same years

# Basics

## Blocks

```
1  {  
2      ...  
3  }
```

- ✓ Good practice to **indent** the code to the right every time a block is open, and to the left when is closed.

## Comments

```
1  //Single line comment  
2  
3  /* Multiple  
4  lines  
5  comments */
```

# Main and libraries

## Libraries

```
1  #include<stdio.h>
2  #include<math.h>
```

- ✓ For including code defined elsewhere.
- ✓ Can be custom libraries or standard libraries like:
  - `stdio.h` stands for *ST*andard *I*ntput *O*utput and provide basic interface with the terminal;
  - `math.h` provide some useful mathematical functions, like `pow` and `sqrt` .
- ✓ The `#include` directives must be write on top of the source file (before the main)

## Main

- ✓ Is the entry point for the program.

```
1  int main(void) {  
2      ...  
3      return 0;  
4  }
```

or

```
1  int main(int argc, char *argv[]) {  
2      ...  
3      return 0;  
4  }
```



# Example

hello\_world.c

```
1 //Compile it with gcc hello_world.c -o
   ↪ hello_world
2
3 #include<stdio.h> //library for input/output
4
5 int main(void) { //begin of main
6     printf("Hello_world!\n"); //output of
   ↪ string
7     return 0;
8 }
```

# Variables

## Initialization

```
1  int var1;           //default initial value
2  float var2 = 3.1415; //custom initial value
3  int var3, var4, var5; //multiple initialization
```

- ✓ C is **case sensitive**, `int`  $\neq$  `Int`  $\neq$  `INT` .
- ✓ Allowed names can contains `[A-Z,a-z,0-9,_]` , cannot begin with a number.
- ✓ Good practices are: to use **camel case** or `_` for composed words, and to start with **lower case**. I.e. `camelCaseExample` .
- ✓ Also when possible define variables on method begin.

# Variables

## Assignment

```
1 var1 = 42;
```

- ✓ Is possible to use also **expressions** on the right of =
- ✓ Also with other variables or the same variable.

```
1 var3 = var4 + var5;  
2 var3 = var3 - 1;
```

# Variable types

## Integer

type	size	min value	max value
char	1 byte	-128	127
short	2 bytes	-32,768	32,767
int	4 bytes	-2,147,483,648	2,147,483,647
long	8 bytes	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
unsigned char	1 byte	0	255
unsigned short	2 bytes	0	65,535
unsigned int	4 bytes	0	4,294,967,295
unsigned long	8 bytes	0	18,446,744,073,709,551,615

✓ char is used also for the characters of the **ascii** table.

# Variable types

## Floating point

type	size	min value	max value	epsilon
float	4 bytes	$1.175494e^{-38}$	$3.402823e^{38}$	$1.192093e^{-07}$
double	8 bytes	$2.225074e^{-308}$	$1.797693e^{308}$	$2.220446e^{-16}$
long double	16 bytes	$3.362103e^{-4932}$	$1.189731e^{4932}$	$1.084202e^{-19}$

## Boolean

- ✓ Is possible to use the type `bool`, and the values `true` and `false`.
- ✓ You need to add `#include<stdbool.h>`.
- ✓ Not really useful because you can use any integer type with the values:
  - 0 for false;
  - $\neq 0$  for true.

# Operators (in priority order)

## Arithmetic

1. `*` , `/` , `%` ;
2. `+` , `-` ;

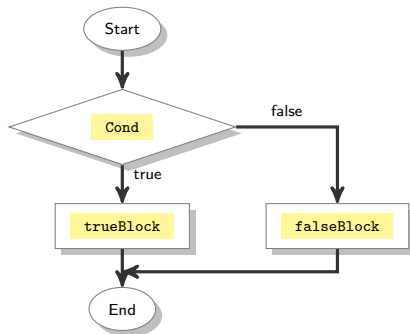
## Logic

2. `!` ;
3. `<` , `>` , `<=` , `>=`
4. `==` , `!=` ;
5. `&&` ;
6. `||` ;

✓ Is possible to change order with `(` and `)` , i.e.:

```
1 (a+b)*c
```

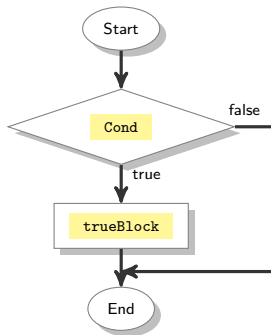
# Selections



```
1  if(cond) {  
2    trueBlock;  
3  } else {  
4    falseBlock;  
5  }
```

# Selections

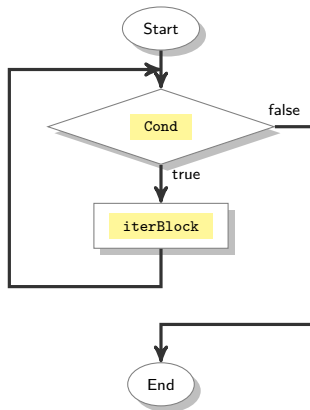
The `else` block is optional.



```
1  if(cond) {  
2      trueBlock;  
3  }
```

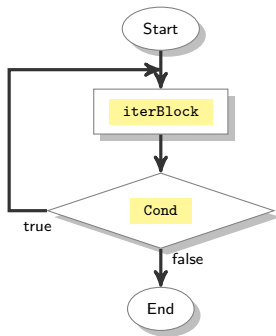


# While iteration



```
1  while(cond) {  
2      iterBlock;  
3  }
```

# Do-while iteration



```
1  do {  
2    iterBlock;  
3  } while(cond);
```

## For iteration

```
1  for(iniz; cond; oper) {  
2      iterBlock;  
3  }
```

is equivalent to:

```
1  iniz;  
2  while(cond) {  
3      iterBlock;  
4      oper;  
5  }
```

## For iteration example

forTest.c

```
1  #include<stdio.h>
2
3  int main(void) {
4      int i; //initialize var
5
6      //iterate from i=1 while i<=42
7      //incrementing i on each loop
8      for(i = 1; i <= 42; i++) {
9          printf("*"); //print an *
10     }
11     printf("\n");
12
13     return 0;
14 }
```

# User input/output

Output: `printf`

```
printf(format, var1, var2, ...);
```

- ✓ `format` is a string that contains the text to be sent to output
- ✓ the format string can contain special chars using `\`
  - `\\` , `\"` , `\%` , `\n` (for new line)
- ✓ the format string can contain special **format specifiers**
  - for each specifier is necessary a corresponding variable `var`
  - the output string will be integrated with the value of the variables

Input: `scanf`

```
scanf(format, &var);
```

- ✓ `format` is a string with only one **format specifier**
- ✓ `var` is a variable name (remember to add the special char `&` )
- ✓ the input will be saved inside `var`

## format specifiers

- ✓ for each **data type** corresponds a **format specifier**
- ✓ a format specifier is `%[length][specifier]`

Length	Specifier		
	d	u	f ( e for $m10^n$ )
(none)	int	unsigned int	float
hh	char	unsigned char	
h	short	unsigned short	
l	long	unsigned long	double
L			long double

For example:

- ✓ `%d` for **int**
- ✓ `%hu` for **unsigned short**
- ✓ `%Le` for **long double** (expressed in scientific notation)

## printfScanfExample.c

```
1  #include <stdio.h>
2
3  int main(void) {
4      float floatVar;
5      long longVar;
6      printf("Input\\output\\ndemonstration\\n\\n");
7      printf("Insert a rational number: ");
8      scanf("%f", &floatVar);
9      printf("Insert a long integer: ");
10     scanf("%ld", &longVar);
11     printf("Rational is: %f; integer is: %ld\\n",
12           ↪ floatVar, longVar);
13     printf("Rational in scientific notation: %e\\n",
14           ↪ floatVar);
15     return 0;
16 }
```

```
1 $ gcc printfScanfExample.c -o example
2 $ ./example
3 Input\output
4 demonstration
5
6 Insert a rational number: 12.34
7 Insert a long integer: 123456
8 Rational is: 12.340000; integer is: 123456
9 Rational in scientific notation: 1.234000e+01
10 $
```