Insper

Mutirão C

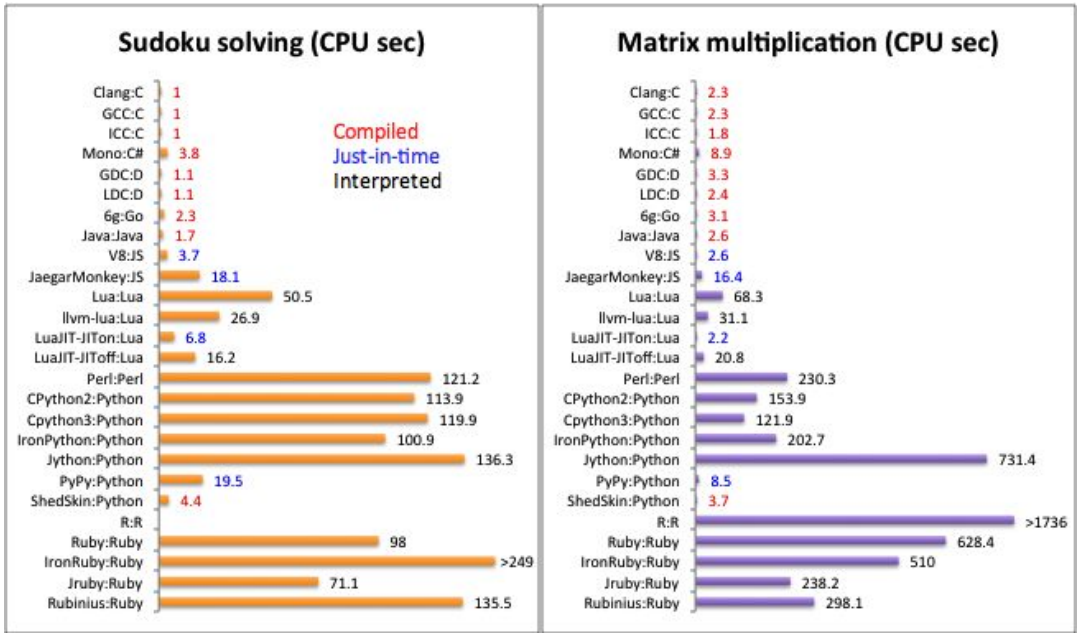
Introdução C
Aula 1

Verdana 12pt

Porque C ?

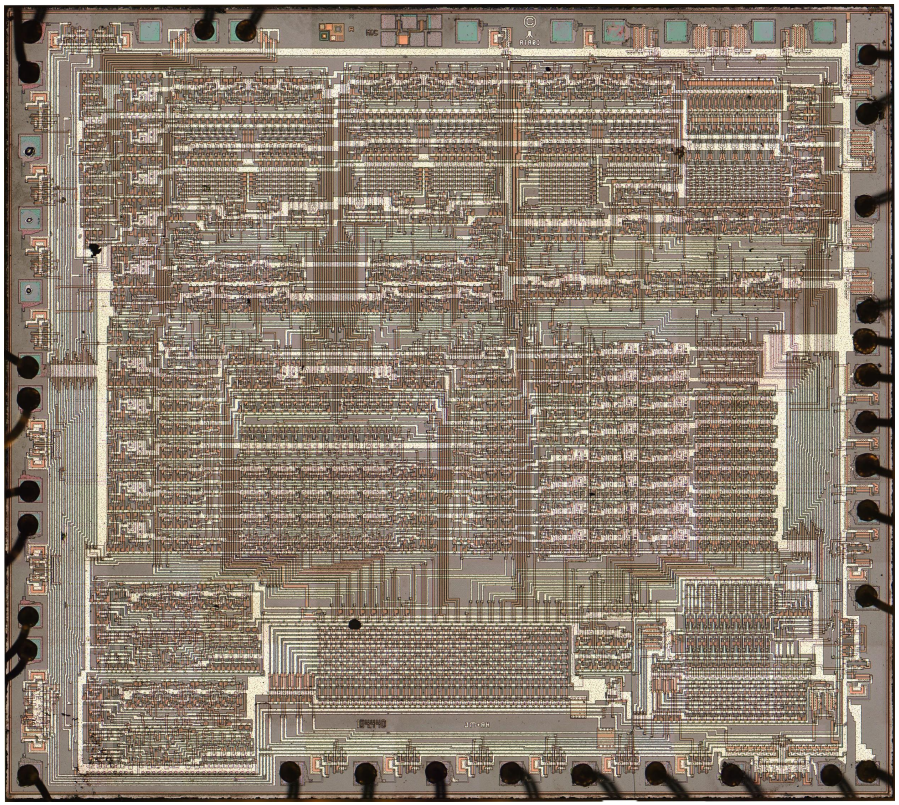
- Possui ótima performance
- Portável
- É bastante utilizado pela comunidade
- Linguagem de “baixo nível“
- Boas ferramentas gratuítas

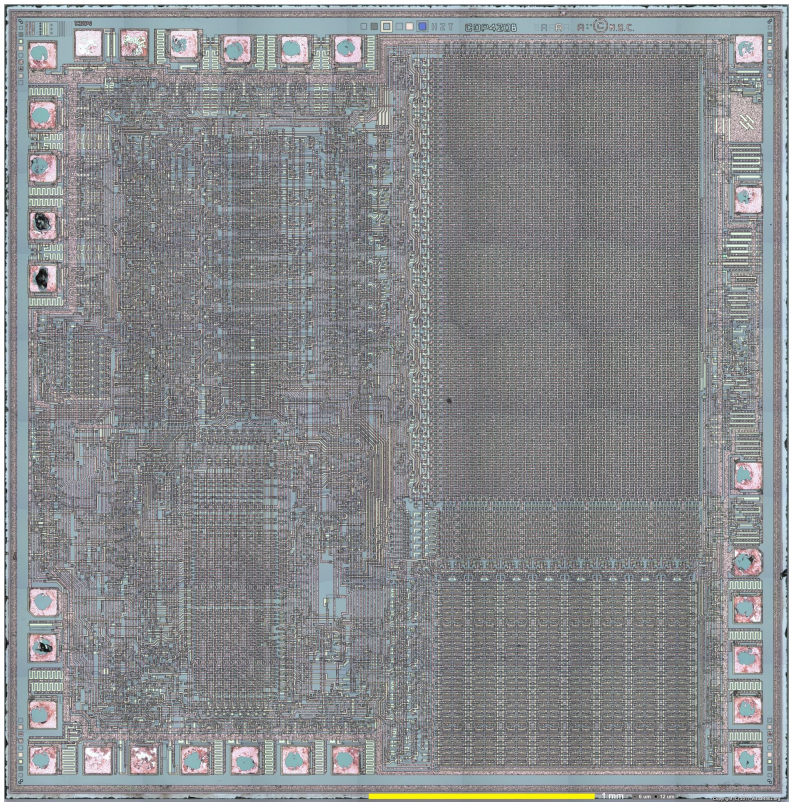
Possui ótima performance



[ref] https://attractivechaos.github.io/plb/

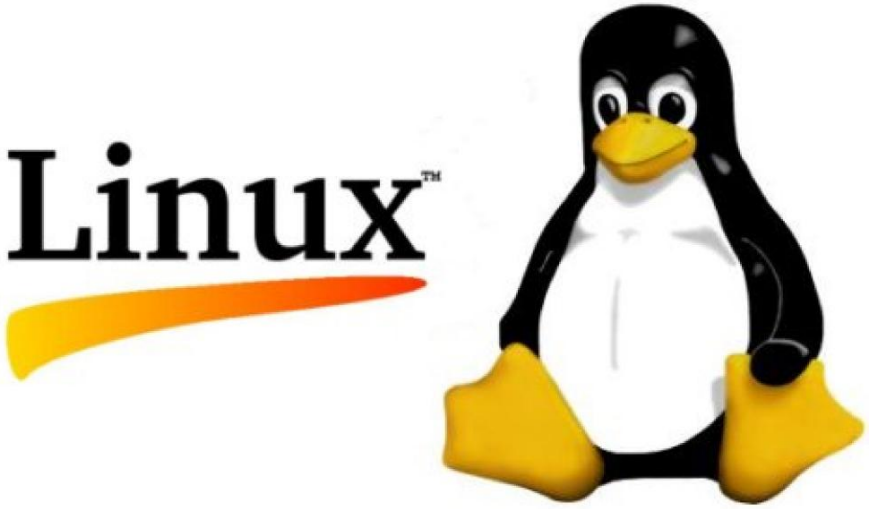
Portável





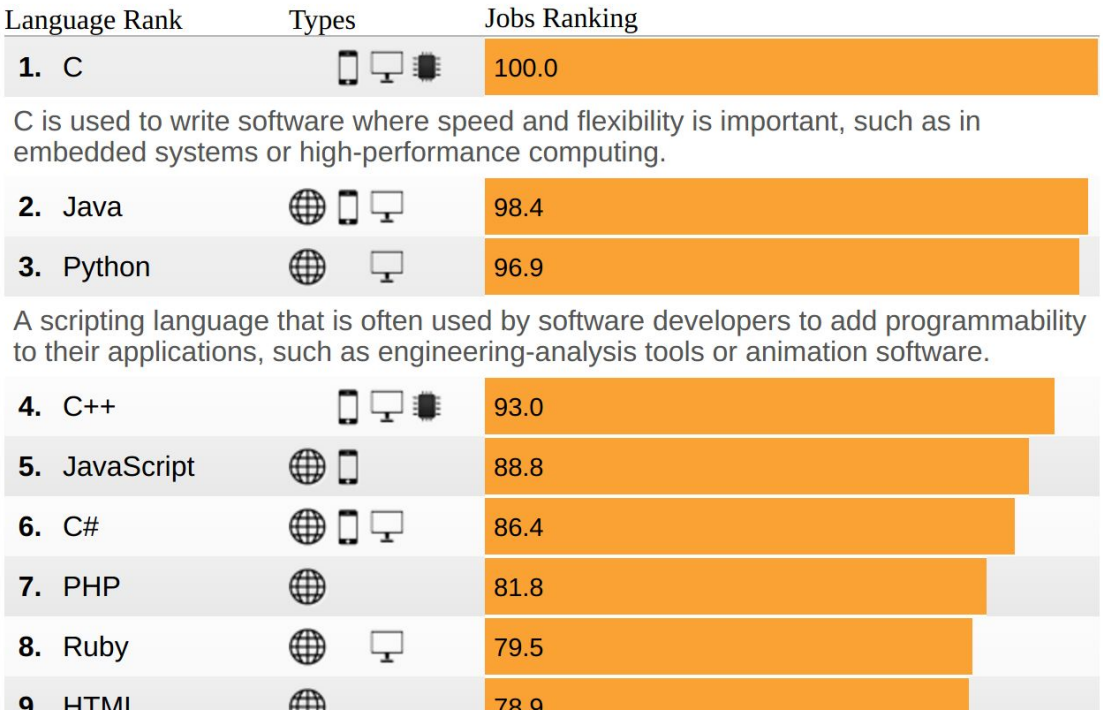
Die : Atari e 8080

Portável



FreeBSD®

É bastante utilizado pela comunidade



Language Rank	Types	Jobs Ranking
1. C		100.0
C is used to write software where speed and flexibility is important, such as in embedded systems or high-performance computing.		
2. Java		98.4
3. Python		96.9
A scripting language that is often used by software developers to add programmability to their applications, such as engineering-analysis tools or animation software.		
4. C++		93.0
5. JavaScript		88.8
6. C#		86.4
7. PHP		81.8
8. Ruby		79.5
9. HTML		78.9

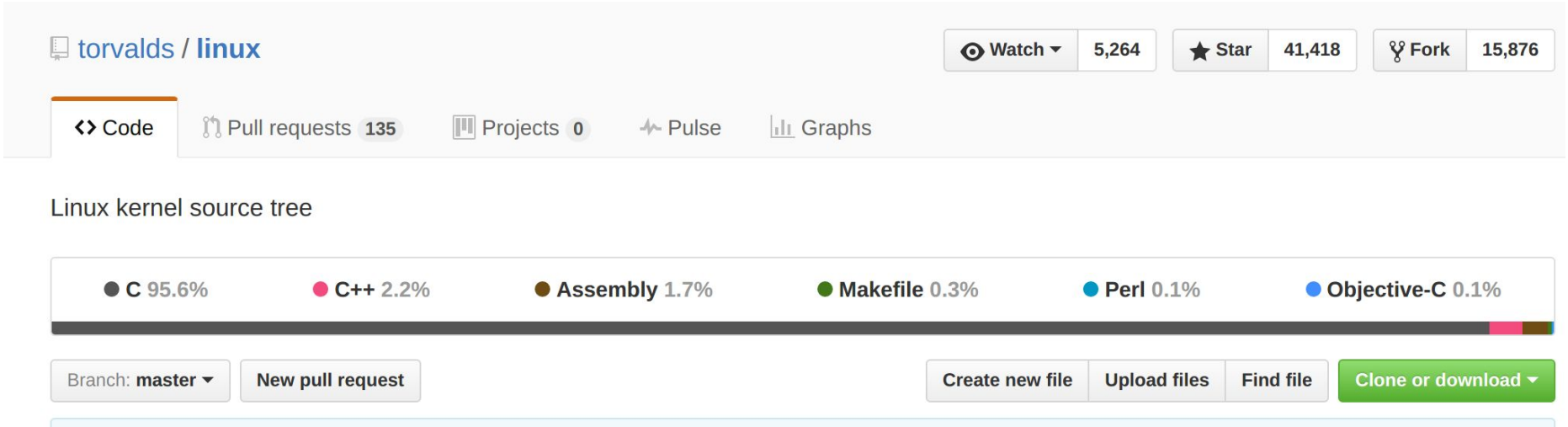
[ref] http://spectrum.ieee.org/static/interactive-the-top-programming-languages-2016

github

[ref] http://githut.info/

Insper

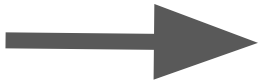
Linux Kernel



[ref] https://github.com/torvalds/linux

Baixo nível

```c
#include <stdio.h>

int main(){

    printf("Hello !! \n");
    return 0;
}
```



```
        .file   "hello.c"
        .section .rodata
.LC0:
        .string "Hello !! "
        .text
        .globl  main
        .type   main, @function
main:
.LFB0:
        .cfi_startproc
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register 6
        movl    $.LC0, %edi
        call    puts
        nop
        popq    %rbp
        .cfi_def_cfa 7, 8
        ret
        .cfi_endproc
.LFE0:
        .size   main, .-main
        .ident  "GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.4) 5.4.0
20160609"
        .section .note.GNU-stack,"",@progbits
```

Programação Aulas

Aula 1 - 8/2	- Introdução C	- Rafael / Fábio
Aula 2 - 9/2	- Funções, vetores, matrizes e strings	- Fábio / Hashi
Aula 3 - 10/2	- Ponteiros	- Hashi / Fábio
Aula 4 - 13/2	- Desafio 1	- Fábio / Hashi
Aula 5 - 13/2	- Desafio 2	- Rafael / Fábio
Aula 6 - 15/2	- Avaliação	- Rafael / Fábio / Hashi

Compilando

Infraestrutura

- Linux ou/
  - VmWare / VirtualBox com Linux
- gcc (`sudo apt-get install build-essential`)
- github
  - (https://github.com/Insper/mutiraoC)

Como um código em C parece ?

```c
#include <stdio.h>

void main(){

    printf("Hello !! \n");

}
```

Bibliotecas a serem utilizadas

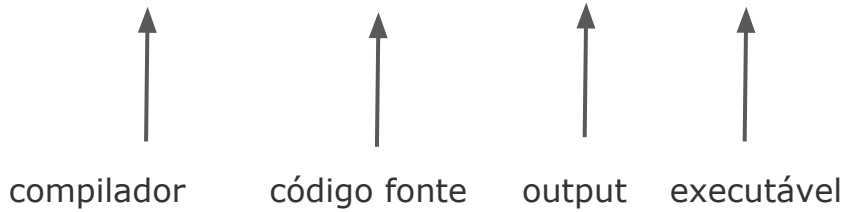
Função principal

Imprime na tela um texto.

mutiraoC/Aula1/hello.c

GCC

```
$ gcc hello.c -o hello
```




```
1.  # Gera o Assembler
2.  gcc hello.c -o hello
3.
4.  # Executa o programa
5.  ./hello
```

Syntax

Syntax ~= Java

- Aritimética :
  - * +,-,*,/,%
  - * ++,--,*=,...
- Relacional: <,>,<=,>=,==,!=
- Lógico: &&, ||, !, ? :
- Bit: &,|,^,!,<<,>>

- if( ){ } else { }
- while( ){ }
- do { } while( )
- for(i=0; i<100; i++){ }
- switch( ) { case 0: ... }
- break, continue, return

Exercício 1 (8 min)

O código **mmc.c** deveria calcular o maior divisor comum (MDC) entre dois números, porém o mesmo apresenta problemas.

Corrija o código e valide o programa.

Tipos

Tipos inteiros

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes	0 to 65,535 or 0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long	4 bytes	0 to 4,294,967,295

https://www.tutorialspoint.com/cprogramming/c_data_types.htm

Tipos inteiros - **Variam** !!!

PIC 18F microcontrolador

TABLE 2-1: INTEGER DATA TYPE SIZES AND LIMITS

Type	Size	Minimum	Maximum
`char`(1,2)	8 bits	-128	127
`signed char`	8 bits	-128	127
`unsigned char`	8 bits	0	255
`int`	16 bits	-32,768	32,767
`unsigned int`	16 bits	0	65,535
`short`	16 bits	-32,768	32,767
`unsigned short`	16 bits	0	65,535
`short long`	24 bits	-8,388,608	8,388,607
`unsigned short long`	24 bits	0	16,777,215
`long`	32 bits	-2,147,483,648	2,147,483,647
`unsigned long`	32 bits	0	4,294,967,295

Note 1: A plain *char* is signed by default.

2: A plain *char* may be unsigned by default via the `-k` command-line option.

[ref] http://ww1.microchip.com/downloads/en/DeviceDoc/C18_User_Guide_51288j.pdf

Insper

Tipos inteiros - x64

x86_64 GNU/Linux

```
  Var       Bytes

char           1
unsigned char  1
signed char    1
int            4
unsigned int   4
short          2
unsigned short 2
long           8
unsigned long  8
```

/exemplos/Aula1/tamanhoTipos.c

http://ww1.microchip.com/downloads/en/DeviceDoc/C18_User_Guide_51288j.pdf

Tipos floating-point

Type	Storage size	Value range	Precision
float	4 byte	1.2E-38 to 3.4E+38	6 decimal places
double	8 byte	2.3E-308 to 1.7E+308	15 decimal places
long double	10 byte	3.4E-4932 to 1.1E+4932	19 decimal places

x86_64 GNU/Linux

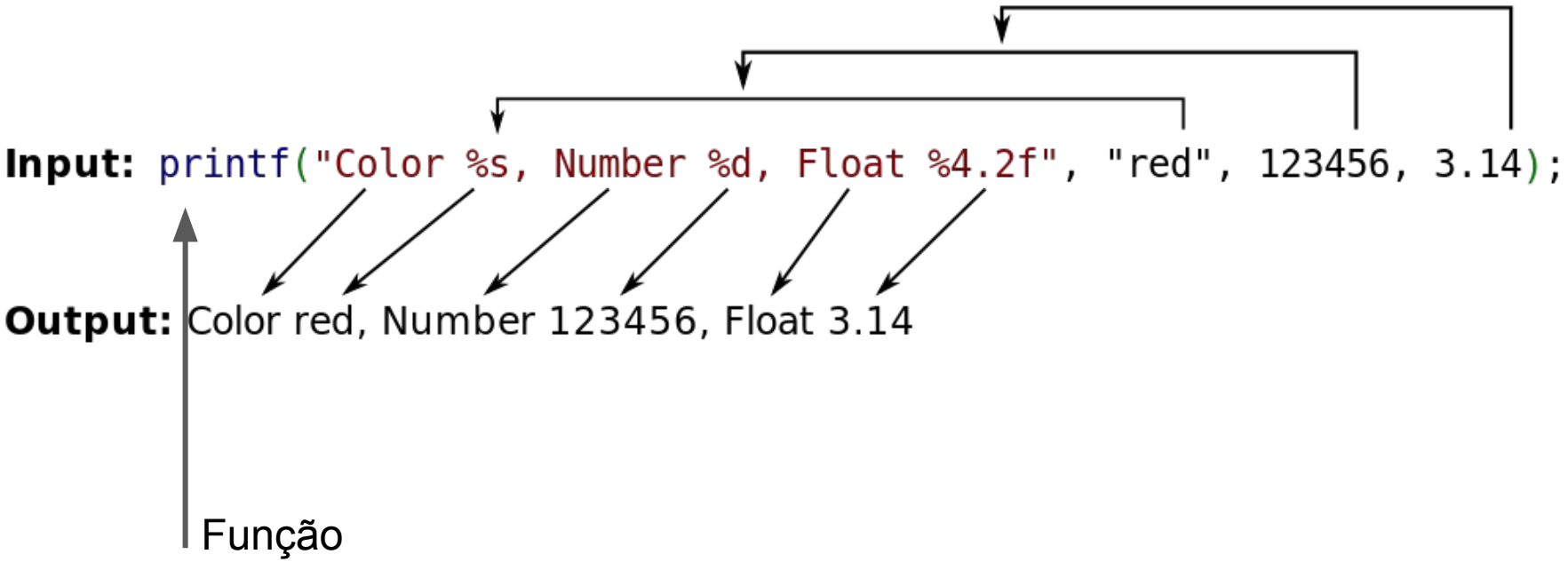
```
  Var        Bytes

long            4
double          8
long double     16
```

← /exemplos/Aula1/tamanhoTipos..c

[ref] https://www.tutorialspoint.com/cprogramming/c_data_types.htm

printf / scanf

printf - exibição de dados



https://en.wikipedia.org/wiki/Printf_format_string

Printf

```c
#include <stdio.h>

void main(){
    unsigned char contador;
    for(contador=0; contador < 10; contador++){
        printf("Val %d \n", contador);
    }
}
```

```
Val 0
Val 1
Val 2
Val 3
Val 4
Val 5
Val 6
Val 7
Val 8
Val 9
```

mutiraoC/Aula1/**printf.c**

Exercício 2 (8 min)

Altere o código **printf.c** para exibir na tela um contador porém agora classificando se o número é ímpar ou par, exemplo :

```
Val 1 (ímpar)
Val 2 (par)
Val 3 (ímpar)
Val 4 (par)
```

(interrompa quando chegar em 1024)

scanf - leitura de dados

/exemplos/Aula1/4-scanf.c

```
#include <stdio.h>

int main(void)
{
    int  n;
    scanf("%d", &n);
    printf("\nO valor inserido foi : %d \n", n);
    return 0;
}
```

```
./4-scanf
12

O valor inserido foi : 12
```

mutiraoC/Aula1/**scanf.c**

Exercício 3 (8 min)

Faça um programa (em C) que calcule o juros compostos de um produto, o programa deve receber como parâmetros :

PV: Valor Presente
i **:** Taxa de Juros
n **:** Número de Períodos

e retornar o :

FV**:** Valor Futuro

Utilize a equação:

FV = PV x (1 + i) ^ n

[ref] http://hcinvestimentos.com/2009/06/21/juros-compostos/?hvid=2BYUMA

Exercicío 4 (para casa ?!)

Escreva um programa que receba do usuário 3 itens (nome, preço e quantidade) e formate o resultado similar a tabela:

Produto	Preço Unitário	Quantidade	Preço Total
Banana	R$ 2.50	2	R$ 5.00
Uva	R$ 6.50	6	R$ 39.00
Pessego	R$ 10.22	10	R$ 102.20
		Sub-Total	R$ 146.20
	Imposto (5%)		R$ 7.31
		Total	R$ 153.51

Bibliografias

http://www.slideshare.net/amraldo/introduction-to-c-programming-7898353

ftp://ftp.ufv.br/dma/tutoriais/c%2B%2B/introd_ling_c.pdf