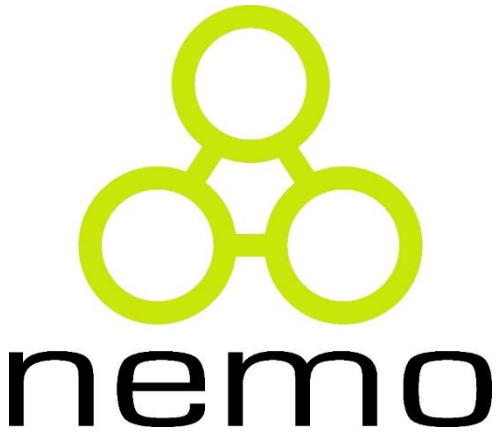


Programação 2

Jordana S. Salamon

jssalamon@inf.ufes.br

DEPARTAMENTO DE INFORMÁTICA

CENTRO TECNOLÓGICO

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

Introdução a Linguagem C

nemo

História

- C é uma linguagem de programação compilada de propósito geral, estruturada, imperativa, procedural, padronizada pela ISO, criada em 1972, por Dennis Ritchie.

Ken Thompson e Dennis Ritchie (da esquerda para direita), os criadores das linguagens B e C, respectivamente.

nemo

Definindo um algoritmo em C

Tradução	
Pseudocódigo	**C**

Algoritmo MeuPrograma
VAR
...
Início
...
Fim.
	main() {
	...
	//comentário
	/*
	bloco de comentário
	*/
	...
	}

- Comentários: entre /* e */ ou depois de //
- *Case sensitive*: main ≠ Main ≠ MaIn ≠ mAin ≠ ...
- Uso de “;” ao final de cada instrução

nemo

Função Main

- É uma função **especial** da linguagem.
- Sempre o arquivo de execução for executado, está função será chamada.

```
int main();

int main (void);

void main ();

main ();

int main (int argc, char *argv[])

int main (int argc, char **argv)
```

nemo

Função Main()

- Podemos utilizar a forma mais simples
  - main(){ }
- Forma completa
  - int main(int argc, char **argv){ }
- Retorno indica sucesso ou falha para o SO
  - Sucesso: = 0
  - Erro: > 0
- Parâmetros
  - Lista de parâmetros (argv)
  - Quantidade de parâmetros (argc)

Variáveis

- Nome dado ao local da memória capaz de armazenar um valor.
- No programa, através do nome da variável é possível acessar o valor (ou conteúdo) que lá está.
- Podemos dizer que uma variável nada mais é do que uma abstração para o endereço de memória.

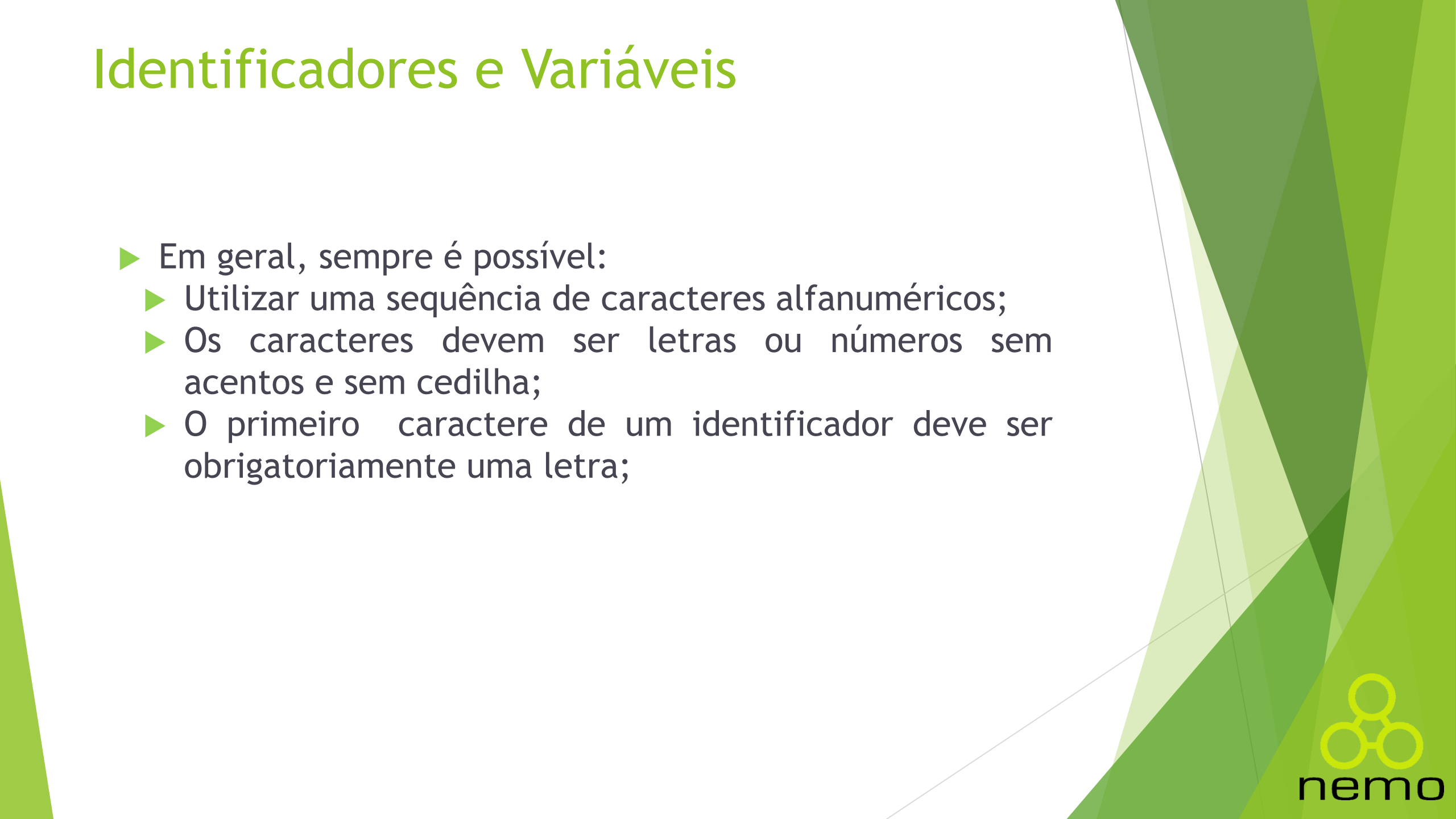
Identificadores

- Em geral, as linguagens de alto nível possuem dois tipos de elementos: os elementos definidos pela própria linguagem (símbolos para operadores, nome de comandos, etc), e os elementos definidos pelo programador (identificadores, comentários, etc);
- Um identificador é um símbolo que pode representar alguma entidade criada pelo programador, como uma **variável** por exemplo;
- Cada linguagem define uma regra para formação de identificadores;

nemo

Identificadores e Variáveis

- Em geral, sempre é possível:
  - Utilizar uma sequência de caracteres alfanuméricos;
  - Os caracteres devem ser letras ou números sem acentos e sem cedilha;
  - O primeiro caractere de um identificador deve ser obrigatoriamente uma letra;



Identificadores e Variáveis

- Exemplos:

```
1    abc
2    x1
3    y2
4    letra
5    SOMA_TOTAL
6    B_32
```

Exemplo 2.1: Nomes válidos de variáveis, concordando com as regras de nomenclatura.

```
1    fim?            // ‘‘?’’ não é um caractere alfanumérico
2    %percentual%    // ‘‘%’’ não é um caractere alfanumérico
3    123quatro       // Iniciado por número
4    !hola!          // ‘‘!’’ não é um caracter alfanumérico
5    @ARROBA         // ‘‘@’’ não é um caractere alfanumérico
```

Exemplo 2.2: Nomes inválidos de variáveis.

nemo

Identificadores e Variáveis

- Algumas linguagens fazem diferenciação entre letras maiúsculas e minúsculas.
- Escolher nomes para as variáveis que sejam intuitivas quanto ao seu uso é uma boa política
- É recomendável adotar padrões para a escrita de identificadores
- Normalmente, em grandes projetos de software, são adotados padrões para a escrita dos identificadores a fim de que os programadores possam trocar seus códigos, entendê-los e alterá-los sem grande dificuldade.

nemo

Identificadores e Variáveis

- Sugestão:
  - **Nomes simples**: começando com letra minúscula e demais caracteres minúsculos;
  - **Nomes compostos**: primeira parte iniciada por letra minúscula e as demais partes iniciadas por letra maiúscula. Os demais caracteres são minúsculos.
- Exemplo:

```
1    delta
2    raiz1
3    idade
4    letra
5    percentualDeLucro
6    primeiraLetra
7    indiceBovespa
```

Exemplo 2.3: Nomes significativos para variáveis.

nemo

Comando de Atribuição

- Serve para alterar os valores (conteúdo) das variáveis.
- Exemplo:

```c
/*
 * main.c
 *
 *  Created on: 20/11/2012
 *      Author: clebson
 */

#include <stdlib.h>

int main()
{
    int leituraAtual = 125;
    int leituraAnterior = 25;
    float valorUnitario = 2.5;
    int diferenca;
    float valorConta;

    diferenca = leituraAtual - leituraAnterior ;
    valorConta = diferenca * valorUnitario ;

    return 0;
}
```

- Qual o valor da variável **valorConta** ao final da execução do programa?

nemo

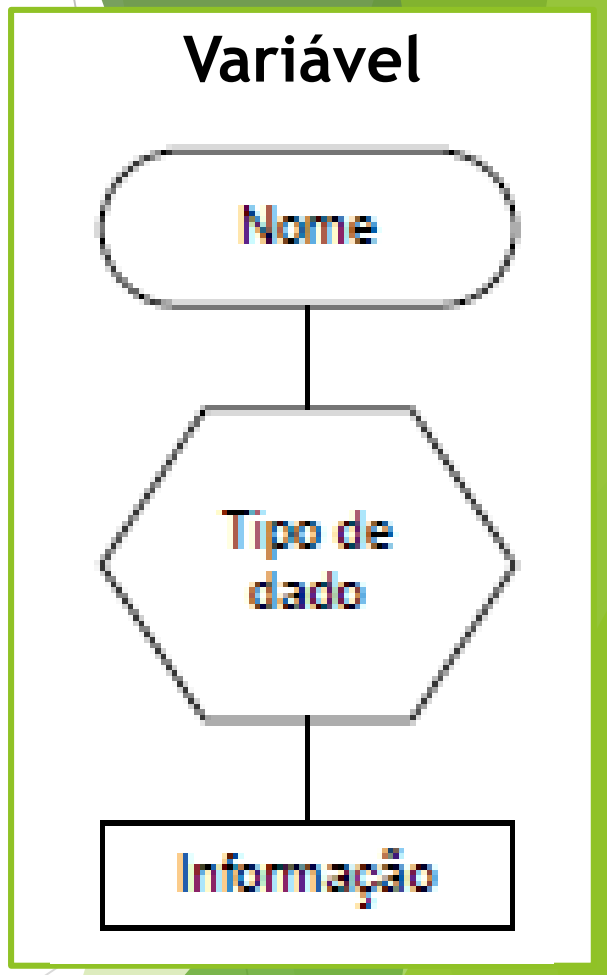
Tipo de Dados

- Um tipo de dado delimita o conjunto de valores possíveis que uma determinada variável pode representar e suas operações básicas;
- São necessários porque uma única célula de memória representa um conjunto de dados muito limitado;
- Tipos de dados são abstrações sobre palavras de memória;
- O tamanho de cada tipo de dado varia com a implementação do compilador/interpretador e com o tipo do processador utilizado;

Tipo de Dados e Declaração de Variáveis

Tradução	
Pseudocódigo	**C**
Inteiro	int
Real	float
Literal	char
Literal[]	char[]
Lógico	int

Exemplos	
Pseudocódigo	**C**
VAR a, b, c: **inteiro** x, y: **real** pagou: **lógico** nome: **literal**[50]	**int** a, b, c; **float** x, y; **int** pagou; **char** nome[50];



- Uso do “;”
- Tamanho do `char`
  - Diferença entre `char` e `char[10]`
- Não existe `booleano` em C: usa-se 0 e 1
- `short`, `long`, `double`, `long double`

nemo

Instruções Primitivas

Tradução	
Pseudocódigo	**C**
<-	=
Leia	scanf()
Escreva	printf()

Exemplos	
Pseudo	**C**
a <- 10	a = 10;
Leia x	scanf("%tipo", &x);
Leia nome	scanf("%s", &nome);
Leia nome, idade	scanf("%s%d", &nome, &idade);
Escreva "Olá, mundo!"	printf("Olá, mundo!\n");
Escreva "Nome: ", nome, " e idade: ", idade	printf("Nome: %s e idade: %d\n", nome, idade);

nemo

Códigos de Leitura e Impressão

Tradução	
Código	**Tipo**
`%d`	`int`
`%f`	`float`
`%c`	`char //único`

- `printf()` não quebra linha automaticamente
  - `\n`
- `scanf()` e `printf()` requerem adição de uma biblioteca
  - `#include <stdio.h>`
- `scanf()` necessita do caractere `&` a cada variável

nemo

Códigos de Leitura e Impressão

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int leituraAtual , leituraAnterior , diferenca ;
    float valorUnitario , valorConta ;

    printf ( "Digite o valor da leitura ATUAL : " ) ;
    scanf ( "%d" , & leituraAtual ) ;

    printf ( "Digite o valor da leitura ANTERIOR : " ) ;
    scanf ( "%d" , & leituraAnterior ) ;

    printf ( "Digite o preco do Quilowatt - hora : " ) ;
    scanf ( "%f" , & valorUnitario ) ;

    diferenca = leituraAtual - leituraAnterior ;
    valorConta = diferenca * valorUnitario ;

    printf ( "Valor da conta R$ : %f", valorConta ) ;


    return 0;
}
```

nemo

Tabela de tipos de Dados Completa

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Inicio	Fim
char	8	%c	-128	127
unsigned char	8	%c	0	255
signed char	8	%c	-128	127
int	16	%i	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i	-32.768	32.767
short int	16	%hi	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi	-32.768	32.767
long int	32	%li	-2.147.483.648	2.147.483.647
signed long int	32	%li	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
float	32	%f	3,4E-38	3.4E+38
double	64	%lf	1,7E-308	1,7E+308
long double	80	%Lf	3,4E-4932	3,4E+4932

Tabela de tipos de Dados Completa

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Inicio	Fim
char	8	%c	-128	127
unsigned char	8	%c	0	255
signed char	8	%c	-128	127
int	16	%i	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i	-32.768	32.767
short int	16	%hi	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi	-32.768	32.767
long int	32	%li	-2.147.483.648	2.147.483.647
signed long int	32	%li	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
float	32	%f	3,4E-38	3.4E+38
double	64	%lf	1,7E-308	1,7E+308
long double	80	%Lf	3,4E-4932	3,4E+4932

Depende da arquitetura do processador!!!!

O tipo long é 32 bits como int em computadores de arquitetura 32 bits e 64 bits em computadores de arquitetura 64 bits

nemo

Tabela de tipos de Dados Completa

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
char	8	%c		
unsigned char	8	%c		
signed char	8	%c		
int	16	%i		
unsigned int	16	%u		
signed int	16	%i		
short int	16	%hi		
unsigned short int	16	%hu		
signed short int	16	%hi		
long int	32	%li	-2	
signed long int	32	%li	-2	
unsigned long int	32	%lu		
float	32	%f		
double	64	%lf		
long double	80	%Lf	3,4E-4932	3,4E+4932

Aqui, apresentamos inteiro com 2 bytes, mas eles podem ter 4 bytes, isso vai depender do processador do computador, ie, com quantos bytes consegue ele lidar ao mesmo tempo.

Segundo o padrão C99, não existe nenhuma garantia de que uma variável short int é menor que uma variável int, nem que long int é maior que int. Apenas é garantido que int não é maior que long nem menor que short. De fato, nos sistemas x86 de 32 bits, o tamanho de int é igual ao de long. Geralmente, int será o tamanho nativo do processador — ou seja, 32 bits num processador de 32 bits, 16 bits num processador de 16 bits etc.

nemo

Tabela de tipos de Dados Completa

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Inicio	Fim
char	8	%c	-128	127
unsigned char	8	%c	0	255
signed char	8	%c	-128	127
int	16	%i	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i	-32.768	32.767
short int	16	%hi	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi	-32.768	32.767
long int	32	%li	-2.147.483.648	2.147.483.647
signed long int	32	%li		
unsigned long int	32	%lu		
float	32	%f		
double	64	%lf	1,7E-308	1,7E+308
long double	80	%Lf	3,4E-4932	3,4E+4932

O tipo long double trabalha em máquinas x64 no padrão LP64 (Mac OS X e Unix)

nemo

Tabela de tipos de Dados Completa

- Resumindo:
- O tamanho depende da arquitetura do processador;
- Os tipos mais utilizados serão char, int, float, double;

Exemplo - Pseudocódigo

Faça um algoritmo que leia uma temperatura em Fahrenheit e calcule sua correspondente em Celsius. Ao fim, imprima os dois valores. [C = (5*(F-32)/9)]

```
Algoritmo Fahrenheit
VAR
    tCelsius, tFahrenheit: real;
Início
    escreva “Informe o valor em Fahrenheit: ”
    leia tFahrenheit
    tCelsius <- (5 * (tFahrenheit - 32)/9)
    escreva “Valor em Celsius: ”, tCelsius
Fim
```

nemo

Exemplo - C

Faça um algoritmo que leia uma temperatura em Fahrenheit e calcule sua correspondente em Celsius. Ao fim, imprima os dois valores. [C = (5*(F-32)/9)]

```c
#include <stdio.h>

void main(){
    float tempC, tempF;
    printf("Informe o valor em Fahrenheit: ");
    scanf("%f", &tempF);
    tempC = (5* (tempF - 32) /9);
    printf("Valor em Celsius: %f\n", tempC);
}
```

nemo

Constantes

- Constantes diferem em relação a variáveis pois não utilizam posições de memória durante a execução do programa;
- Definição de constantes em C:
  - **#define <identificador> <valor>**
- Exemplo:

```
 2  * main.c
 3  *
 4  *  Created on: 20/11/2012
 5  *      Author: clebson
 6  */
 7
 8 /*Utilizacao de constantes*/
 9
10 #include <stdlib.h>
11
12 # define PI 3.141593
13 # define FALSO 0
14 # define VERDADEIRO 1
15
16 int main()
17 {
18     float area = 0, raio = 3.5;
19
20     raio = PI*raio*raio;
21
22     return 0;
23 }
```

nemo

Expressões Aritméticas

Tradução						
Pseudo	+	-	*	/	^	√
C	+	-	*	/	pow()	sqrt()

Exemplos	
Pseudo	**C**
`d <- √(a + (b - c) * 4) / (2^10)`	`d = sqrt(a + (b - c) * 4) / pow(2, 10);`
`c <- c + 1`	`c = c + 1;`
`c <- c + 1`	`c++;`
`c <- c + 3`	`c += 3;`
`c <- c - 1`	`c = c - 1;`
`c <- c - 1`	`c--;`
`c <- c - 3`	`c -= 3;`

nemo

Expressões Lógicas

Tradução					
Pseudo	.V.	.F.	.OU.	.E.	.NÃO.
C	1	0	||	&&	!

Exemplos	
Pseudo	**C**
pagou .E. maiorDeIdade	pagou && maiorDeIdade

nemo

Expressões Relacionais

Tradução						
Pseudo	=	<>	<	<=	>	>=
C	==	!=	<	<=	>	>=

Exemplos	
Pseudo	C
`(a = b) .OU. (.NÃO. (b = c))`	`(a == b) || (! (b == c))`
`(x > y) .E. (z < w)`	`(x > y) && (z < w)`
`resposta <> 10`	`resposta != 10`

nemo

Condicionais: Se-Então

Tradução	
Pseudocódigo	**C**

Se <Condição> Então
  <Conjunto de Comandos 1>
Senão
  <Conjunto de Comandos 2>
Fim_se
	if (<Condição>){
	  <Comandos 1>;
	}else{
	  <Comandos 2>;
	}

Se <Condição> Então
  <Conjunto de Comandos>
Fim_se
	if (<Condição>){
	  <Comandos>;
	}

nemo

Exemplo de IF

```c
#include <stdio.h>

main() {
    float N1, N2, Media;
    printf("Informe a Nota 1: ");
    scanf("%f", &N1);
    printf("Informe a Nota 2: ");
    scanf("%f", &N2);
    Media = (N1 + N2) / 2;
    if(Media >= 7){
        printf("Aprovado\n");
    }else{
        printf("Reprovado\n");
    }
}
```

nemo

Exercício

- Construa um algoritmo que receba a idade do usuário e informe se ele tem idade maior ou igual a 18 anos ou não.

```
main() {
   ...
}
```

```
if (<Condição>){
   <Comandos 1>;
}else{
   <Comandos 2>;
}
```

```
if (<Condição>){
   <Comandos>;
}
```

```
int a, b, c;
float x, y;
int pagou;
```

```
scanf("%tipo", &x);
```

```
printf("Idade: %d\n", idade);
```

nemo

Resolução

```c
#include <stdio.h>

main(){
    int idade;
    printf("Digite a idade: ");
    scanf("%d", &idade);
    if(idade >= 18){
        printf("Maior de idade, com %d anos", idade);
    } else{
        printf("Menor de idade, com %d anos", idade);
    }
}
```

nemo

Desvio Condicional: Else if

- Em certas situações quando temos múltiplas condições, utilizar apenas if e elses pode tornar o código complexo.
- A linguagem C oferece alguns recursos para facilitar a tomada de decisão em múltiplas escolhas, um deles é o else if.
- Exemplo: Dado 2 números (A e B) informe se A > B, ou B > A, ou se eles são iguais.

nemo

Desvio Condicional: Else if

```c
#include <stdio.h>

main(){
    int a, b;

    printf("Digite o valor de A e B: \n");
    scanf("%d %d", &a, &b);

    if(a>b){
        printf("A maior que B.\n");
    } else{
        if(a<b){
            printf("A menor que B.\n");
        }
        else{
            printf("A igual a B.\n");
        }
    }
}
```

nemo

Desvio Condicional: Else if

```c
#include <stdio.h>

main(){
    int a, b;

    printf("Digite o valor de A e B: \n");
    scanf("%d %d", &a, &b);

    if(a>b){
        printf("A maior que B.\n");
    }
    else if (a<b){
        printf("A menor que B.\n");
    }
    else{
        printf("A igual a B.\n");
    }
}
```

nemo

Desvio Condicional: Switch

- Seleção múltipla:

```
switch (<expressão>)
{
    case <valor1>: <sequência de comandos 1>
                   break;
    case <valor2>: <sequência de comandos 2>
                   break;
    ...
    ...
    case <valorN>: <sequência de comandos N>
                   break;
    default: <sequência de comandos>
}
```

Desvio Condicional: Switch

```
switch (num) {
  case 10:
    printf("Igual a 10.\n");
    break;
  case 5:
    printf("Igual a 5.\n");
    break;
  default:
    printf("Nem 10 nem 5.\n");
}
```

nemo

Desvio Condicional: Switch

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int numero ;

    printf ("URNA ELETRONICA - SEU VOTO PARA PREFEITO : " );
    scanf ("%d", &numero);

    switch(numero)
    {
        case 1: printf("Candidato escolhido : Hortencia da Silva.\n");
                break;
        case 2: printf("Candidato escolhido : Jose dos Cravos.\n");
                break;
        case 3: printf("Candidato escolhido : Margarida S. Pereira.\n");
                break;
        default:printf("Numero digitado invalido. Voto anulado.\n");
                break;
    }

    return 0;
}
```

That’s all Folks!