atec

Algoritmos
UFCD 0804 – 1.2

Nelson Santos
nelson.santos.0001376@edu.atec.pt

Agenda

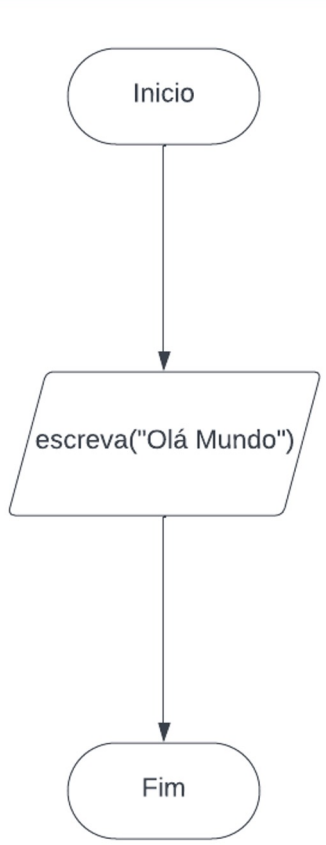
- Variáveis e constantes
- Tipos de dados

Conceitos genéricos

Resumo da aula anterior

Primeiro exercício

```
1  programa
2  {
3      funcao inicio ()
4      {
5          escreva("Olá Mundo")
6      }
7  }
8
9
```



Variáveis

Binário/Decimal

- Para que seja possível armazenar e manipular dados no computador é necessário representá-los internamente de alguma forma. Por norma, nós seres humanos, representamos os números através de um sistema que chamamos de sistema decimal (ou sistema na base 10).
- Esse sistema, que teve origem no fato de utilizarmos os 10 dedos das mãos, possui 10 dígitos distintos para representar as infinitas quantidades e valores que desejamos (012345678e9).

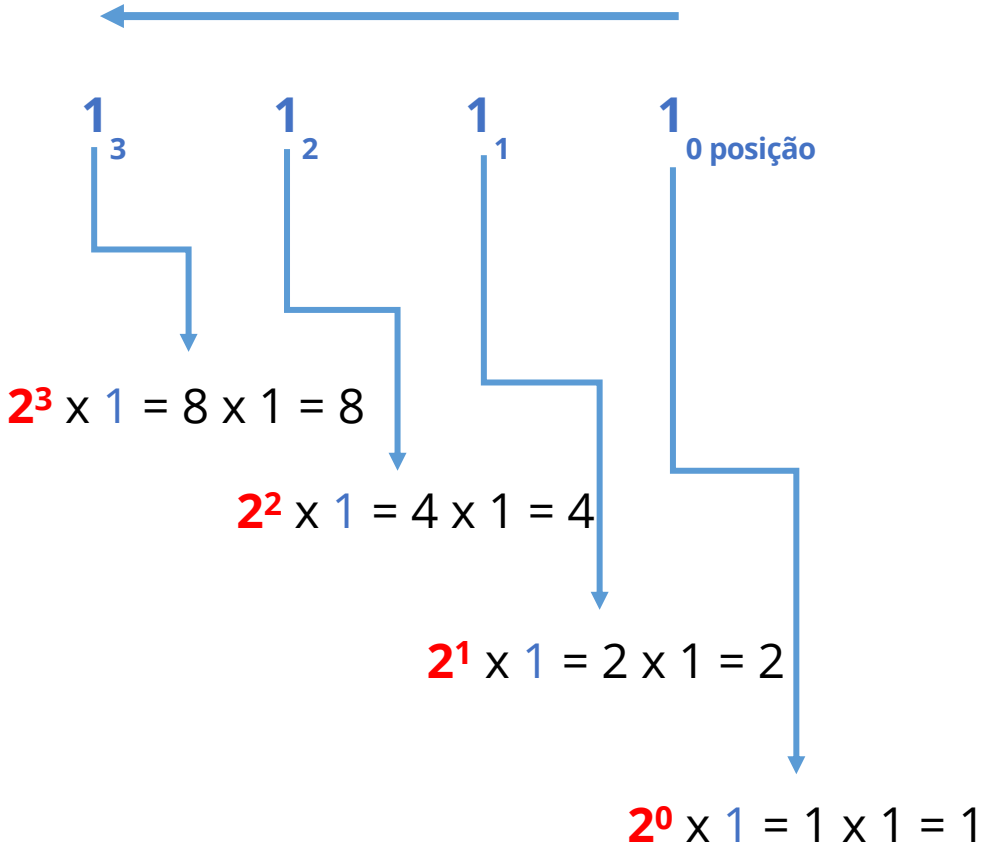
Variáveis

Explicação

- Nos caso dos computadores digitais, a notação que é utilizada possui apenas 2 algarismos ou dígitos para representar uma quantidade desejada, o 0 e o 1
- Este sistema de representação é chamado de sistema binário (ou sistema na base 2) e utiliza uma notação binária (0/1 , verdadeiro/falso, sim/não)
- Outras formas de representação auxiliares também são utilizadas nos computadores, como por exemplo a representação pelo sistema hexadecimal (ou sistema na base 16) que utiliza 16 dígitos (0 1 2 3 4 5 6 7 8 9 A B C D E F)

Variáveis

Exercícios Binário/Decimal – método usando as posições

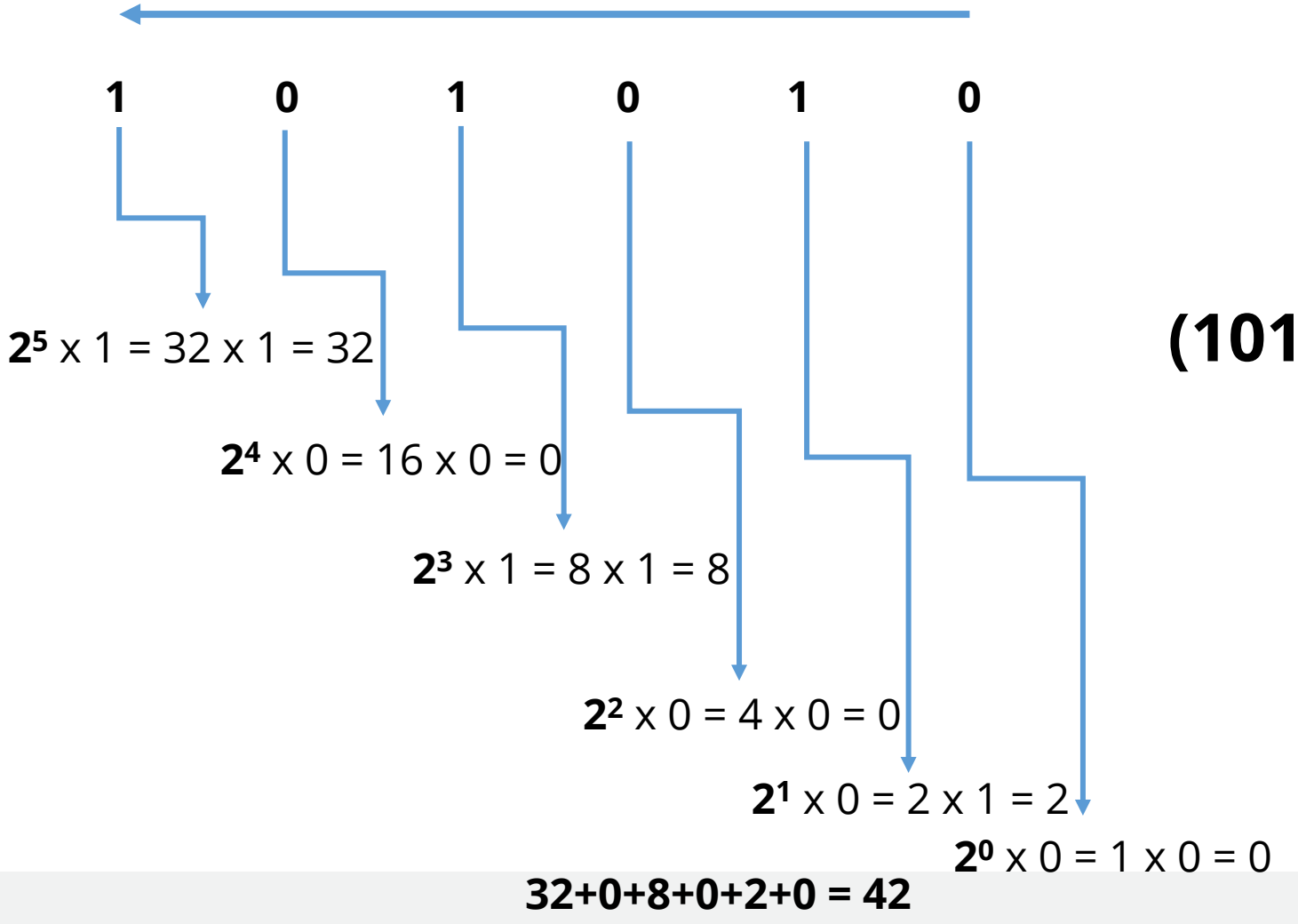


$(1111)_2 = (15)_{10}$

8 + 4 + 2 + 1 = 15

Variáveis

Exercícios Binário/Decimal – método usando as posições



$$(101010)_2 = (42)_{10}$$

Variáveis

Exercícios Binário/Decimal – método usando as posições

- 101?
- 1010?
- 100000000?

Variáveis

Exercícios Binário/Decimal – método usando as posições

- **101**
  - $(101)_2 = (5)_{10}$
- **1010**
  - $(1010)_2 = (10)_{10}$
- **100000000**
  - $(100000000)_2 = (256)_{10}$

Variáveis

Exercícios Decimal/Binário – método usando as posições

Base	Número a dividir	Resto
2	17	1
2	8	0
2	4	0
2	2	0
	1	

17	2
1	8

17/2 = 8 | resto 1
8/2 = 4 | resto 0
4/2 = 2 | resto 0
2/2 = 1 | resto 0

$(17)_{10} = (10001)$

Variáveis

Exercícios Decimal/Binário – método usando as posições

Base	Número a dividir	Resto
2	25	1
2	12	0
2	6	0
2	3	1
2	1	

25/2 = 12 | resto 1
12/2 = 6 | resto 0
6/2 = 3 | resto 0
3/2 = 1 | resto 1

$(25)_{10}$ = (11001)

Variáveis

Exercícios Decimal/Binário – método usando as posições

- 248?
- 575?
- $(56)_{10}$?

Variáveis

Exercícios Decimal/Binário – método usando as posições

- 248?
  - $(248)_{10} = (11111000)_2$
- 575?
  - $(575)_{10} = (1000111111)_2$
- $(56)_{10}$?
  - $(56)_{10} = (111000)_2$

Variáveis

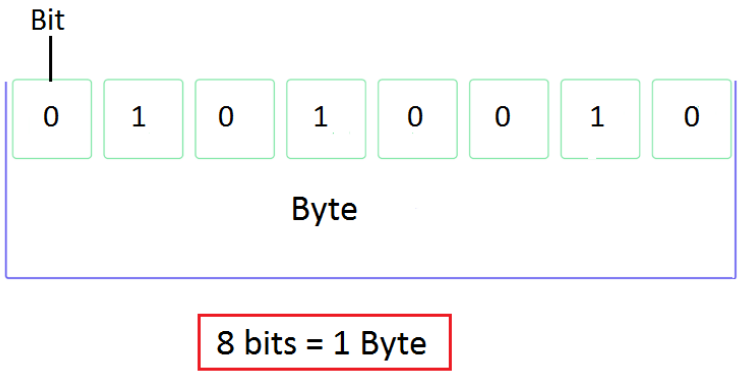
Decimal/hexadecimal/binário

Hexadecimal	Binary	Decimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Bits & Bytes

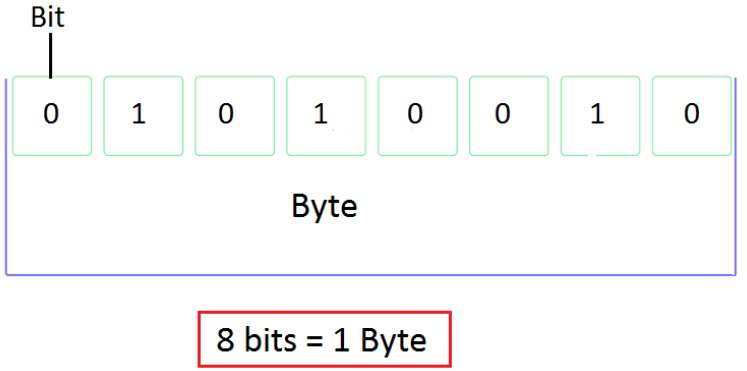
Explicação

- Um "bit" é atómico: a unidade mais pequena de armazenamento
- O bit apenas tem valor 1 ou 0
- “Para o computador é tudo 0's e 1's" ... Bits
- Um bit é demasiado pequeno para ser usado
- Grupos de 8 bits fazem um byte



Bits & Bytes

Explicação



- Um byte pode guardar um character: “A”, “1”, “$”
- Combinações possíveis?

Número de bits	Combinações possíveis
1	0 ou 1
2	00 ou 01 ou 10 ou 11
4	000 ou 001 ou 010 ou 011 ou 100 ou 101 ou 110 ou 111
n	2^n combinações possíveis

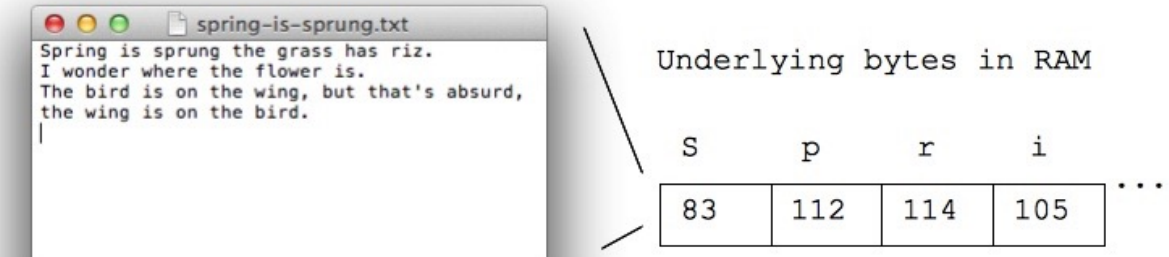
Bits & Bytes

Explicação

Dec	Hex	Binary	Char	Dec	Hex	Binary	Char	Dec	Hex	Binary	Char	Dec	Hex	Binary	Char
0	0x00	00 00000	NUL	32	0x20	01 00000	SPACE	64	0x40	10 00000	@	96	0x60	11 00000	`
1	0x01	00 00001	SOH	33	0x21	01 00001	!	65	0x41	10 00001	A	97	0x61	11 00001	a
2	0x02	00 00010	STX	34	0x22	01 00010	"	66	0x42	10 00010	B	98	0x62	11 00010	b
3	0x03	00 00011	ETX	35	0x23	01 00011	#	67	0x43	10 00011	C	99	0x63	11 00011	c
4	0x04	00 00100	EOT	36	0x24	01 00100	$	68	0x44	10 00100	D	100	0x64	11 00100	d
5	0x05	00 00101	ENQ	37	0x25	01 00101	%	69	0x45	10 00101	E	101	0x65	11 00101	e
6	0x06	00 00110	ACK	38	0x26	01 00110	&	70	0x46	10 00110	F	102	0x66	11 00110	f
7	0x07	00 00111	BEL	39	0x27	01 00111	'	71	0x47	10 00111	G	103	0x67	11 00111	g
8	0x08	00 01000	BS	40	0x28	01 01000	(	72	0x48	10 01000	H	104	0x68	11 01000	h
9	0x09	00 01001	HT	41	0x29	01 01001	)	73	0x49	10 01001	I	105	0x69	11 01001	i
10	0x0a	00 01010	LF	42	0x2a	01 01010	*	74	0x4a	10 01010	J	106	0x6a	11 01010	j
11	0x0b	00 01011	VT	43	0x2b	01 01011	+	75	0x4b	10 01011	K	107	0x6b	11 01011	k
12	0x0c	00 01100	FF	44	0x2c	01 01100	,	76	0x4c	10 01100	L	108	0x6c	11 01100	l
13	0x0d	00 01101	CR	45	0x2d	01 01101	-	77	0x4d	10 01101	M	109	0x6d	11 01101	m
14	0x0e	00 01110	SO	46	0x2e	01 01110	.	78	0x4e	10 01110	N	110	0x6e	11 01110	n
15	0x0f	00 01111	SI	47	0x2f	01 01111	/	79	0x4f	10 01111	O	111	0x6f	11 01111	o
16	0x10	00 10000	DLE	48	0x30	01 10000	0	80	0x50	10 10000	P	112	0x70	11 10000	p
17	0x11	00 10001	DC1	49	0x31	01 10001	1	81	0x51	10 10001	Q	113	0x71	11 10001	q
18	0x12	00 10010	DC2	50	0x32	01 10010	2	82	0x52	10 10010	R	114	0x72	11 10010	r
19	0x13	00 10011	DC3	51	0x33	01 10011	3	83	0x53	10 10011	S	115	0x73	11 10011	s
20	0x14	00 10100	DC4	52	0x34	01 10100	4	84	0x54	10 10100	T	116	0x74	11 10100	t
21	0x15	00 10101	NAK	53	0x35	01 10101	5	85	0x55	10 10101	U	117	0x75	11 10101	u
22	0x16	00 10110	SYN	54	0x36	01 10110	6	86	0x56	10 10110	V	118	0x76	11 10110	v
23	0x17	00 10111	ETB	55	0x37	01 10111	7	87	0x57	10 10111	W	119	0x77	11 10111	w
24	0x18	00 11000	CAN	56	0x38	01 11000	8	88	0x58	10 11000	X	120	0x78	11 11000	x
25	0x19	00 11001	EM	57	0x39	01 11001	9	89	0x59	10 11001	Y	121	0x79	11 11001	y
26	0x1a	00 11010	SUB	58	0x3a	01 11010	:	90	0x5a	10 11010	Z	122	0x7a	11 11010	z
27	0x1b	00 11011	ESC	59	0x3b	01 11011	;	91	0x5b	10 11011	[	123	0x7b	11 11011	{
28	0x1c	00 11100	FS	60	0x3c	01 11100	<	92	0x5c	10 11100	\	124	0x7c	11 11100	|
29	0x1d	00 11101	GS	61	0x3d	01 11101	=	93	0x5d	10 11101	]	125	0x7d	11 11101	}
30	0x1e	00 11110	RS	62	0x3e	01 11110	>	94	0x5e	10 11110	^	126	0x7e	11 11110	~
31	0x1f	00 11111	US	63	0x3f	01 11111	?	95	0x5f	10 11111	_	127	0x7f	11 11111	DEL

Bits & Bytes

Explicação



10100111110000..

Dec	Hex	Binary	Char	Dec	Hex	Binary	Char	Dec	Hex	Binary	Char	Dec	Hex	Binary	Char
0	0x00	00 00000	NUL	32	0x20	01 00000	SPACE	64	0x40	10 00000	@	96	0x60	11 00000	`
1	0x01	00 00001	SOH	33	0x21	01 00001	!	65	0x41	10 00001	A	97	0x61	11 00001	a
2	0x02	00 00010	STX	34	0x22	01 00010	"	66	0x42	10 00010	B	98	0x62	11 00010	b
3	0x03	00 00011	ETX	35	0x23	01 00011	#	67	0x43	10 00011	C	99	0x63	11 00011	c
4	0x04	00 00100	EOT	36	0x24	01 00100	$	68	0x44	10 00100	D	100	0x64	11 00100	d
5	0x05	00 00101	ENQ	37	0x25	01 00101	%	69	0x45	10 00101	E	101	0x65	11 00101	e
6	0x06	00 00110	ACK	38	0x26	01 00110	&	70	0x46	10 00110	F	102	0x66	11 00110	f
7	0x07	00 00111	BEL	39	0x27	01 00111	'	71	0x47	10 00111	G	103	0x67	11 00111	g
8	0x08	00 01000	BS	40	0x28	01 01000	(	72	0x48	10 01000	H	104	0x68	11 01000	h
9	0x09	00 01001	HT	41	0x29	01 01001	)	73	0x49	10 01001	I	105	0x69	11 01001	i
10	0x0a	00 01010	LF	42	0x2a	01 01010	*	74	0x4a	10 01010	J	106	0x6a	11 01010	j
11	0x0b	00 01011	VT	43	0x2b	01 01011	+	75	0x4b	10 01011	K	107	0x6b	11 01011	k
12	0x0c	00 01100	FF	44	0x2c	01 01100	,	76	0x4c	10 01100	L	108	0x6c	11 01100	l
13	0x0d	00 01101	CR	45	0x2d	01 01101	-	77	0x4d	10 01101	M	109	0x6d	11 01101	m
14	0x0e	00 01110	SO	46	0x2e	01 01110	.	78	0x4e	10 01110	N	110	0x6e	11 01110	n
15	0x0f	00 01111	SI	47	0x2f	01 01111	/	79	0x4f	10 01111	O	111	0x6f	11 01111	o
16	0x10	00 10000	DLE	48	0x30	01 10000	0	80	0x50	10 10000	P	112	0x70	11 10000	p
17	0x11	00 10001	DC1	49	0x31	01 10001	1	81	0x51	10 10001	Q	113	0x71	11 10001	q
18	0x12	00 10010	DC2	50	0x32	01 10010	2	82	0x52	10 10010	R	114	0x72	11 10010	r
19	0x13	00 10011	DC3	51	0x33	01 10011	3	83	0x53	10 10011	S	115	0x73	11 10011	s
20	0x14	00 10100	DC4	52	0x34	01 10100	4	84	0x54	10 10100	T	116	0x74	11 10100	t
21	0x15	00 10101	NAK	53	0x35	01 10101	5	85	0x55	10 10101	U	117	0x75	11 10101	u
22	0x16	00 10110	SYN	54	0x36	01 10110	6	86	0x56	10 10110	V	118	0x76	11 10110	v
23	0x17	00 10111	ETB	55	0x37	01 10111	7	87	0x57	10 10111	W	119	0x77	11 10111	w
24	0x18	00 11000	CAN	56	0x38	01 11000	8	88	0x58	10 11000	X	120	0x78	11 11000	x
25	0x19	00 11001	EM	57	0x39	01 11001	9	89	0x59	10 11001	Y	121	0x79	11 11001	y
26	0x1a	00 11010	SUB	58	0x3a	01 11010	:	90	0x5a	10 11010	Z	122	0x7a	11 11010	z
27	0x1b	00 11011	ESC	59	0x3b	01 11011	;	91	0x5b	10 11011	[	123	0x7b	11 11011	{
28	0x1c	00 11100	FS	60	0x3c	01 11100	<	92	0x5c	10 11100	\	124	0x7c	11 11100	|
29	0x1d	00 11101	GS	61	0x3d	01 11101	=	93	0x5d	10 11101	]	125	0x7d	11 11101	}
30	0x1e	00 11110	RS	62	0x3e	01 11110	>	94	0x5e	10 11110	^	126	0x7e	11 11110	~
31	0x1f	00 11111	US	63	0x3f	01 11111	?	95	0x5f	10 11111	_	127	0x7f	11 11111	DEL

Variáveis

Explicação

- Dentro de um algoritmo podemos encontrar basicamente duas classes diferentes de dados, os dados **constantes** e os **variáveis**.
- Um dado é uma **constante** quando seu valor não se altera ao longo do tempo em que o algoritmo é executado, ou seja, permanece o mesmo desde o inicio até ao final da execução.
- Por outro lado, um dado que pode ter o seu valor a alterar durante a execução do programa é designado de **variável**.

Variáveis

Explicação

- O computador armazena os dados que são utilizados nos programas e algoritmos na memória de trabalho ou memória RAM (*Random Access Memory*). A memória do computador é sequencial e dividida em posições. Cada posição de memória permite armazenar uma palavra (conjunto de bytes) de informação e possui um número que indica o seu endereço.
- No fundo, são espaços alocados na memória que recebe um nome (identificador) e tem um tipo onde se armazena um valor
- As variáveis podem ser entendidas como sendo apelidos para as posições de memória
- Vamos supor que queremos fazer um programa que solicita para um usuário digitar a sua idade e exibe a ele quantos anos faltam para ele atingir 100 anos de idade. Precisaremos armazenar a idade do usuário para depois realizar o cálculo 100 - idade_usuario e depois armazenar também o resultado

Variáveis

Explicação/Exemplo

Exemplo de Sintaxe

```
01. caracter nome_variavel
02. inteiro variavel_inicializada = 42
03. real nome_variavel2
04. logico nome_variavel3
05. // ou para declarar varias variáveis de um mesmo tipo:
06. cadeia var1,var2,var3,var4
07. logico var4,var5,var6
```

São espaços alocados na memória que recebe um **nome** (identificador) e tem um **tipo** (inteiro, etc), onde se armazena um **valor**.

Variáveis

Exemplo

```
1   programa
2   {
3       //variável global do tipo inteiro
4       inteiro variavel
5
6       funcao inicio()
7       {
8           //variável local do tipo inteiro
9           inteiro outra_variavel
10
11          //variável local do tipo real já inicializada
12          real altura = 1.79
13
14          cadeia frase = "Isso é uma variável do tipo cadeia"
15
16          caracter inicial = 'P'
17
18          logico exemplo = verdadeiro
19
20          //Imprime 1.79, valor obtido na variável altura
21          escreva(altura)
22      }
23  }
24
```

São espaços alocados na memória que recebe um **nome** (identificador) e tem um **tipo** (inteiro, etc), onde se armazena um **valor**.

Constantes

Explicação

- Existem algumas situações em que precisamos que um determinado parâmetro não tenha seu valor alterado durante a execução do programa. Para isso, existem as constantes. Constante é um identificador cujo valor associado não pode ser alterado pelo programa durante a sua execução
- Para declarar uma constante basta adicionar a palavra reservada **const** seguida do tipo de dado, pelo nome da constante e atribuir um valor a ela.

Constantes

Exemplo

```
programa
{
    //Constante global do tipo de dado real
    const real ACELERACAO_GRAVIDADE = 9.78

    funcao inicio()
    {
        //Vetor constante local do tipo de dado caracter
        const caracter VOGAIS[5] = {'a','e','i','o','u'}

        //Matriz constante local do tipo de dado inteiro
        const inteiro TECLADO_NUMERICO[][] = {{1,2,3},{4,5,6},{7,8,9}}
    }
}

```


Variáveis/Constantes

Boas práticas

Regras para os nomes das variáveis e constantes:

- Os nomes das variáveis e constantes devem representar o que será guardado dentro dela;
- O primeiro caracter de um nome deverá ser sempre alfabético.
- Não podem ser colocados espaços em branco no nome de variáveis ou constantes, usar o UNDERSCORE “_”;
- Não podem utilizar palavras reservadas (inteiro, real, função, etc);
- As constantes são usualmente em LETRA MAIÚSCULA;

Variáveis/Constantes

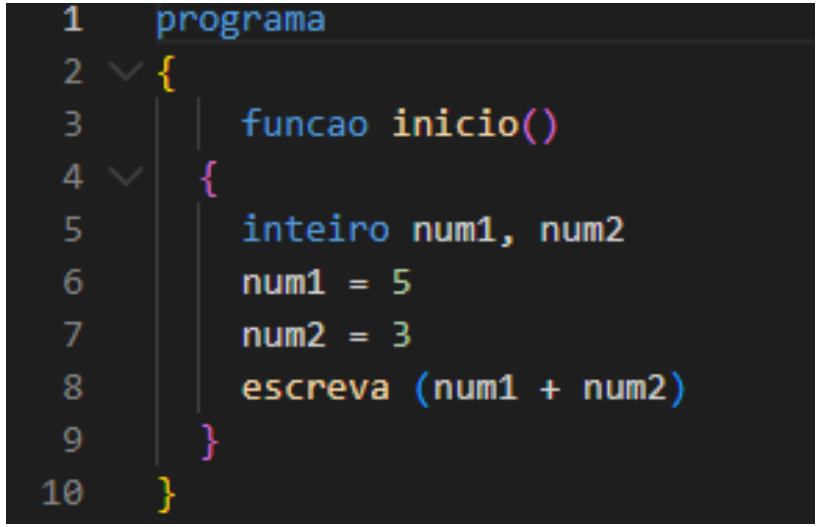
Boas práticas


```
Bad Variable Names to Avoid

// Avoid Single Letter Names
let n = 'use name instead'

// Avoid Acronyms
let cra = 'no clue what this is'

// Avoid Abbreviations
let cat = 'cat or category??'

// Avoid Meaningless Names
let foo = 'what is foo??'
```

Tipos das variáveis

Tipos primitivos

- **Inteiro:** São os números pertencentes ao conjunto dos Inteiros, isto é, que não possuem parte fracionária. Podem ser positivos, nulos ou negativos.
    - Exemplos: 2 laranjas, calçado tamanho 42, 65535 grãos, 0 pessoas na fila, multa de - 2 pontos no campeonato.



```
1  programa
2  {
3      funcao inicio()
4      {
5          inteiro num1, num2
6          num1 = 5
7          num2 = 3
8          escreva (num1 + num2)
9      }
10 }
```

Tipos das variáveis

Tipos primitivos

- **Real:** São os números pertencentes ao conjunto dos Reais, isto é, que podem possuir parte fracionária. Também são chamados de ponto flutuante devido a maneira como o computador os armazena.
    - Exemplos: 2.12 litros de gasóleo, −3.5◦ C, π = 3.141592654, saldo de 10000.52, e = 2.7182818284590451.

```
1   programa
2   {
3       funcao inicio()
4       {
5           real div
6
7           div = 8.0/3.0
8
9           escreva (div)
10      }
11  }
12
```

Tipos das variáveis

Tipos primitivos

- **Caracter:** São os valores pertencentes ao conjunto de todos os caracteres numéricos (0...9), alfabéticos (a...z, A...Z) e especiais (! @ # $ % &).
- Este conjunto também é conhecido como o conjunto de caracteres **alfanuméricos**. Os caracteres alfanuméricos são armazenados internamente no computador na forma numérica (binário) utilizando o padrão ASC□□.

```
1   programa
2   {
3       funcao inicio()
4       {
5       caracter vogal, consoante
6       vogal = 'a'                //variável declarada através de atribuição do programador
7
8       escreva ("Digite uma consoante: ")
9       leia (consoante)            //variável declarada através de entrada do usuário
10
11      escreva ("Vogal: ", vogal, "\n", "Consoante: ", consoante)
12      }
13  }
14
```

Tipos das variáveis

Tipos primitivos

- **Lógico:** O tipo lógico é utilizado para representar informações que só podem assumir dois valores, o valor verdadeiro (V) ou o valor falso (F). Estes valores também podem ser entendidos como: ligado/desligado, 1/0, alto/baixo, fechado/aberto, verdadeiro/falso.
    - Exemplos de informações que podem ser representadas utilizando o tipo lógico são: O fogão está apagado, a televisão está ligada, o portão está aberto, o produto foi encontrado?

```
programa
{
    funcao inicio()
    {
        logico teste
        inteiro num

        escreva ("Digite um valor para ser comparado :")
        leia (num)

        teste = (num>0)

        escreva ("O número digitado é maior que zero? ", teste)
    }
}
```

nput e output

```
programa
{
    funcao inicio()
    {
    inteiro idade
    real salario, nota1, nota2, nota3
    cadeia nome, sobrenome

    escreva("Informe a sua idade: ")
    leia (idade)        //lê o valor digitado para "idade"

    escreva("Informe seu salario: ")
    leia (salario)        //lê o valor digitado para "salario"

    escreva("Informe o seu nome e sobrenome: ")
    leia (nome, sobrenome)    //lê o valor digitado para "nome" e "sobrenome"

    escreva("Informe as suas três notas: ")
    leia (nota1, nota2, nota3)  //lê o valor digitado para "nota1", "nota2" e "nota3"

    escreva("Seu nome é:"+nome+" "+sobrenome+"\n")
    escreva("Você tem "+idade+" anos e ganha de salario "+salario+"\n")
    escreva("Suas três notas foram:\n")
    escreva("Nota 1: "+nota1+"\n")
    escreva("Nota 2: "+nota2+"\n")
    escreva("Nota 3: "+nota3+"\n")
    }
}

```

Operações aritméticas

Operação	Símbolo	Prioridade
Adição	+	1
Subtração	-	1
Multiplicação	*	2
Divisão	/	2
Resto da divisão inteira	%	2

Primeiro são executadas as operações de **prioridade mais elevada** e depois **da esquerda para a direita**

3 + 4 / 2 = **5**

3 * 4 / 2 = **6**

3 + 4 - 2 = **5**

5 % 3 = **2**

1 ** 3 * 4 = **4**

(1 + 3) * 4 = **16**

Ordem	Operação
1ª	Parênteses
2ª	Potenciação ** No Portugol é uma função
3ª	Multiplicação, Divisão, Resto e Divisão Inteira
4ª	Adição, Subtração

Operações aritméticas

Tabela de compatibilidade de tipos da operação de multiplicação

Operando Esquerdo	Operando Direito	Tipo Resultado	Exemplo	Resultado
inteiro	inteiro	inteiro	6 * 8	48
inteiro	real	real	4 * 1.11	4.44
real	inteiro	real	6.712 * 174	1167.888
real	real	real	207.65 * 1.23	255.4095

Tabela de compatibilidade de tipos da operação de divisão

Operando Esquerdo	Operando Direito	Tipo Resultado	Exemplo	Resultado
inteiro	inteiro	inteiro	5 / 2	2
inteiro	real	real	125 / 4.5	27.777777
real	inteiro	real	785.4 / 3	261.8
real	real	real	40.351 / 3.12	12.9333333

Operações aritméticas

Tabela de compatibilidade de tipos da operação de subtração

Operando Esquerdo	Operando Direito	Tipo Resultado	Exemplo	Resultado
inteiro	inteiro	inteiro	20 - 10	10
inteiro	real	real	90 - 0.5	89.5
real	inteiro	real	11.421 - 3	8.421
real	real	real	12.59 - 24.59	-12.0

Tabela de compatibilidade de tipos da operação de módulo

Operando Esquerdo	Operando Direito	Tipo Resultado	Exemplo	Resultado
inteiro	inteiro	inteiro	45 % 7	3

Operações aritméticas

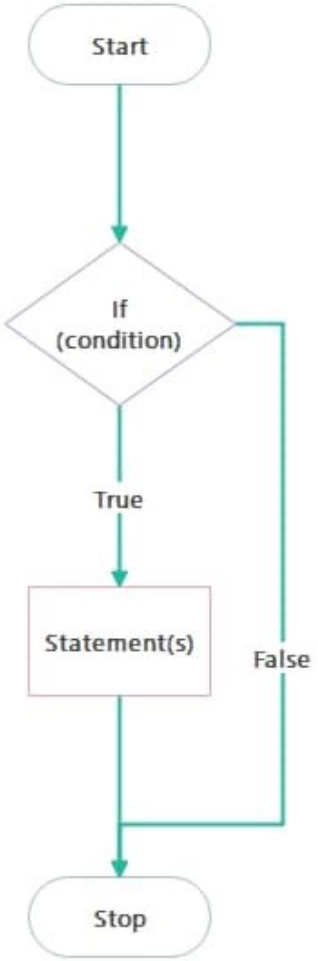
Tabela de compatibilidade de tipos da operação de adição

Operando Esquerdo	Operando Direito	Tipo Resultado	Exemplo	Resultado
cadeia	cadeia	cadeia	"Oi" + " mundo"	"Oi mundo"
cadeia	caracter	cadeia	"Banan" + 'a'	"Banana"
cadeia	inteiro	cadeia	"Faz um" + 21	"Faz um 21"
cadeia	real	cadeia	"Altura: " + 1.78	"Altura: 1.78"
cadeia	logico	cadeia	"Help bom =" + verdadeiro	"Help bom = verdadeiro"
caracter	cadeia	cadeia	'P' + "anqueca"	"Panqueca"
caracter	caracter	cadeia	'C' + 'a' + 'd' + 'e' + 'i' + 'a'	"Cadeia"
inteiro	cadeia	cadeia	22 + " de agosto"	"22 de agosto"
inteiro	inteiro	inteiro	12 + 34	46
inteiro	real	real	76 + 3.25	79.25
real	cadeia	cadeia	3.24 + " Kg"	"3.24 Kg"
real	inteiro	real	9.87 + 1	10.87
real	real	real	9.87 + 0.13	10.0
logico	cadeia	cadeia	verdadeiro + " amigo"	"verdadeiro amigo"

Exercício 1

- Elabore um algoritmo no Portugol e respetivo diagrama onde se solicita duas variáveis reais (r e r1), duas inteiras (i e i2). Depois escreva no ecrã:
  - A soma dos dois reais;
  - A soma dos dois inteiros;
  - A divisão de r por i;
  - A divisão de i2 por i;
  - O módulo de i por i2;
  - A divisão da soma dos números reais por i2;
  - A multiplicação de todos os números;

Decisão condicional simples




```
1   programa
2   {
3       funcao inicio()
4       {
5
6           inteiro num
7
8           escreva ("Digite um número: ")
9           leia (num)
10
11          se (num==0)
12          {
13              escreva ("O número digitado é 0")
14          }
15
16      }
17  }
18
```

Exercício 2

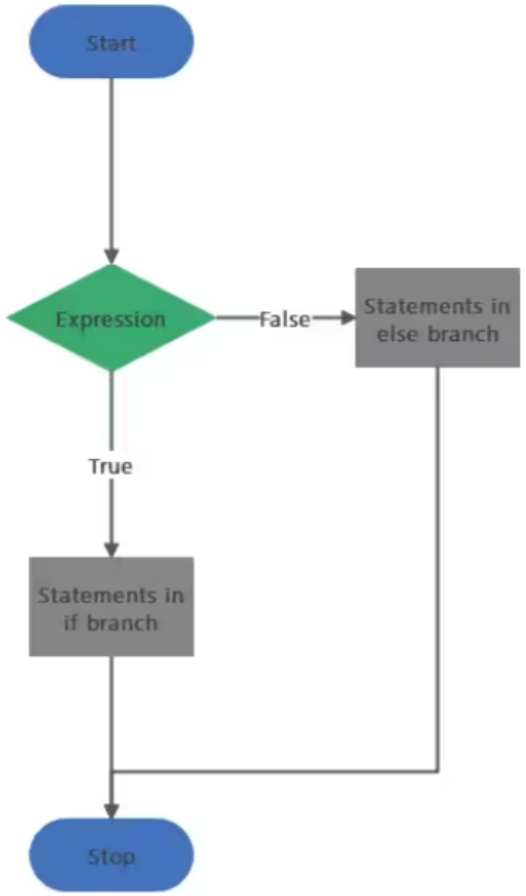
- Elabore um algoritmo no Portugol que solicite um número inteiro ao utilizador e diga se é par. Caso não seja, não escreve nada. Fazer o diagrama no *flowchart*.

$$\text{even} \equiv 0 \pmod 2$$

$$\text{odd} \equiv 1 \pmod 2$$

Par = 0 se mod 2 => número % 2 = 0 ? É par
Impar = 1 se mod 2 => número % 2 = 1 ? É impar
Operador resto da divisão: %

Decisão condicional composta




```
1   programa
2   {
3     funcao inicio()
4     {
5
6       inteiro hora
7
8       escreva ("Digite a hora: ")
9       leia (hora)
10
11      se (hora >= 6 e hora <= 18)
12      {
13        escreva ("É dia")
14      }
15      senao
16      {
17        escreva ("É noite")
18      }
19
20    }
21  }
22
```

Exercício 3

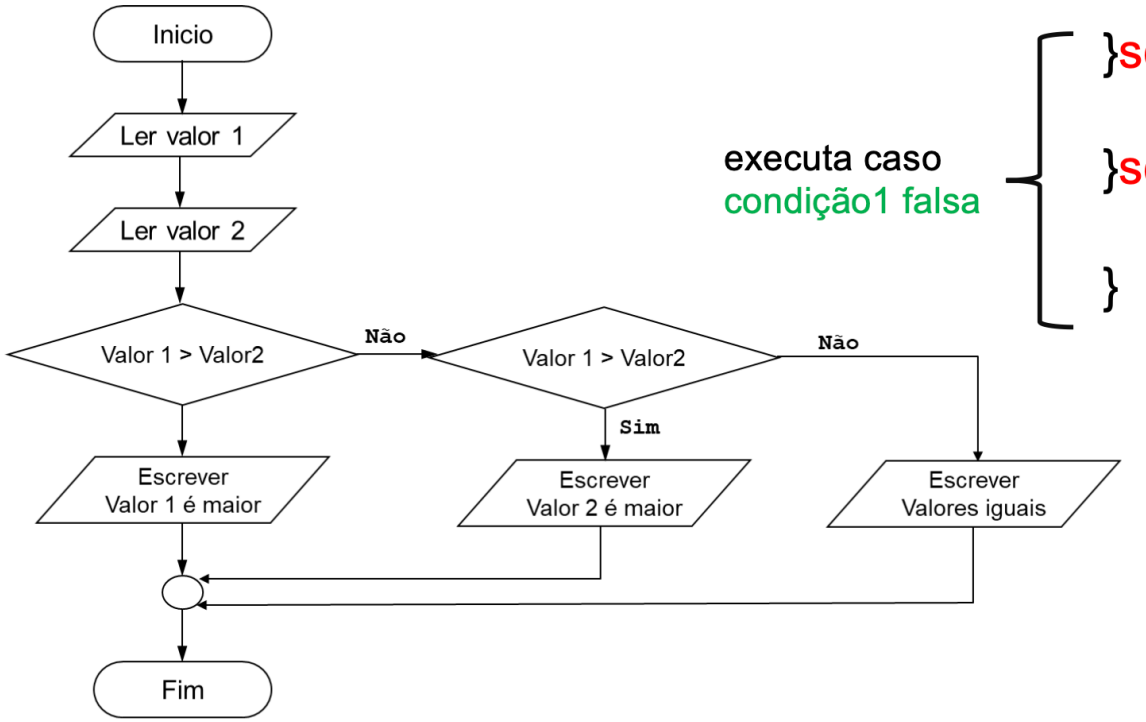
- Elabore um algoritmo no Portugol que solicite um número inteiro ao utilizador e diga se é par ou ímpar. Fazer o diagrama no *flowchart*.

$$\text{even} \equiv 0 \pmod 2$$

$$\text{odd} \equiv 1 \pmod 2$$

Par = 0 se mod 2 => número % 2 = 0 ? É par
Ímpar = 1 se mod 2 => número % 2 = 1 ? É impar
Operador resto da divisão: %

Decisão condicional composta 2

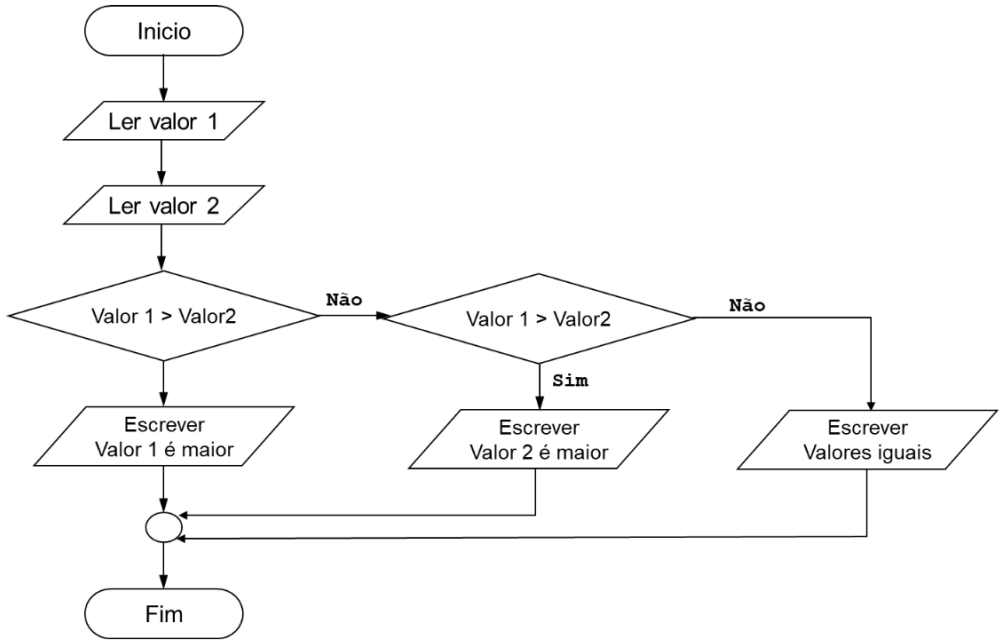


executa caso
condição1 falsa

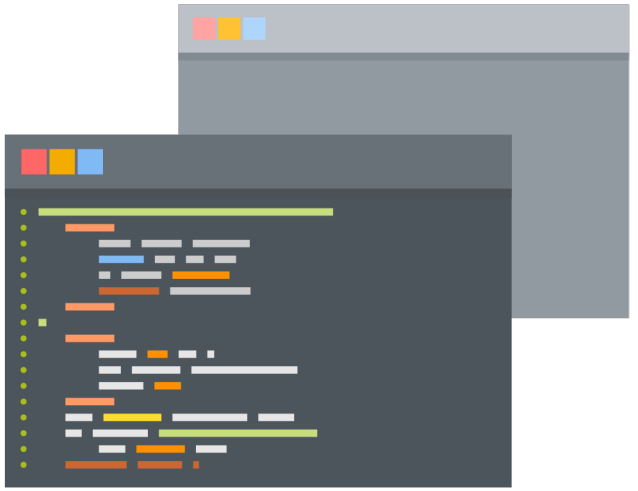
```
se(condição1){
    executa caso condição1 verdadeira
}senao se(condição2){
    executa caso condição2 verdadeira
}senao{
    executa caso condição2 falsa
}
```

Exercício 4

- Elabore um algoritmo no Portugol que solicite ao utilizador dois números inteiros e escreva se o primeiro número é maior, menor ou igual que o segundo.



3 Questões



PALMELA
Edifício ATEC · Parque Industrial da Volkswagen Autoeuropa
2950-557 · Quinta do Anjo
Tel. 212 107 300 | info@atec.pt

PORTO
Edifício Siemens · Av. Mário Brito (EN107), nº 3570 · Freixieiro
4456-901· Perafita
Tel. 220 400 500 | infoporto@atec.pt

www.atec.pt