

FUNDAMENTOS DE PROGRAMAÇÃO ORIENTADA A OBJETO

Profº Ms Gustavo Molina

Prof. Ms Gustavo Molina



- Graduado em Sistemas de Informação pelo MACKENZIE.
- Licenciado em Matemática pela UNIP.
- Pós – Graduado em Plataforma de Desenvolvimento Web pelo CLARETIANO.
- Pós – Graduado em IA pela faculdade Serra Geral
- Pós – Graduado em Gestão e Governança de Tecnologia da Informação pela UNIP
- Mestre em Engenharia Elétrica pela FEI
- Doutorando em Ciências da Educação pela Ivy Enber Christian University

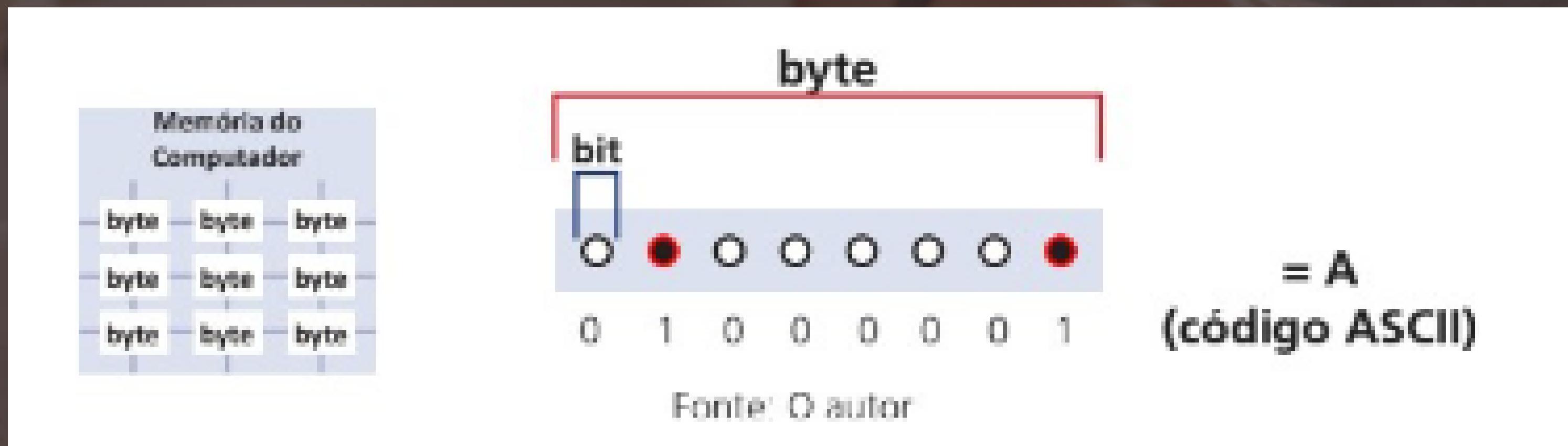


Aula 4

Byte

A menor unidade utilizável para representação de informações em um computador é o BIT, que assume os valores 0 ou 1.

Na figura a seguir, vemos a memória do computador e o BIT/BYTE.



Quando nos referimos às informações armazenadas em um computador, utilizamos, portanto, o termo **byte**, que corresponde a um caractere (letra, número ou símbolo). Tendo em vista que a unidade byte é consideravelmente pequena quando indicamos valores mais extensos, utilizamos múltiplos do byte: **kilobyte**, **megabyte**, **gigabyte**, **terabyte**, etc.

CARACTER / STRING

- Conforme conversamos, os bytes representam todas as letras (maiúsculas e minúsculas), sinais de pontuação, acentos, CARACTERES especiais e até informações que não podemos ver, mas que servem para comandar o computador.
- STRING também é conhecido como um conjunto de caracteres em algumas linguagens de programação, como por exemplo, Java e Python.

O nome ASCII vem do inglês *American Standard Code for Information Interchange* ou "Código Padrão Americano para o Intercâmbio de Informação". Ele é baseado no alfabeto romano e sua função é padronizar a forma como os computadores representam letras, números, acentos, sinais diversos e alguns códigos de controle

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	'	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	,	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	:	91	5B	[123	7B	[
28	1C	[FILE SEPARATOR]	60	3C	\	92	5C	\	124	7C	\
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D]
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	-	127	7F	[DEL]

Um PROGRAMA DE COMPUTADOR ou simplesmente SOFTWARE é representado por instruções e dados que alguém definiu para ser executado e cumprir com o seu objetivo



```
1 import java.util.Scanner;
2 class PerimeterOfRectangle
3 {
4     public static void main(String args[])
5     {
6
7         Scanner s= new Scanner(System.in);
8
9         System.out.println("Enter the length of the Rectangle:");
10
11        double l= s.nextDouble();
12
13        System.out.println("Enter the width of the Rectangle:");
14
15        double b= s.nextDouble();
16
17        double perimeter=2*(l+b);
18
19        System.out.println("perimeter of Rectangle is: " + perimeter);
20    }
21 }
```

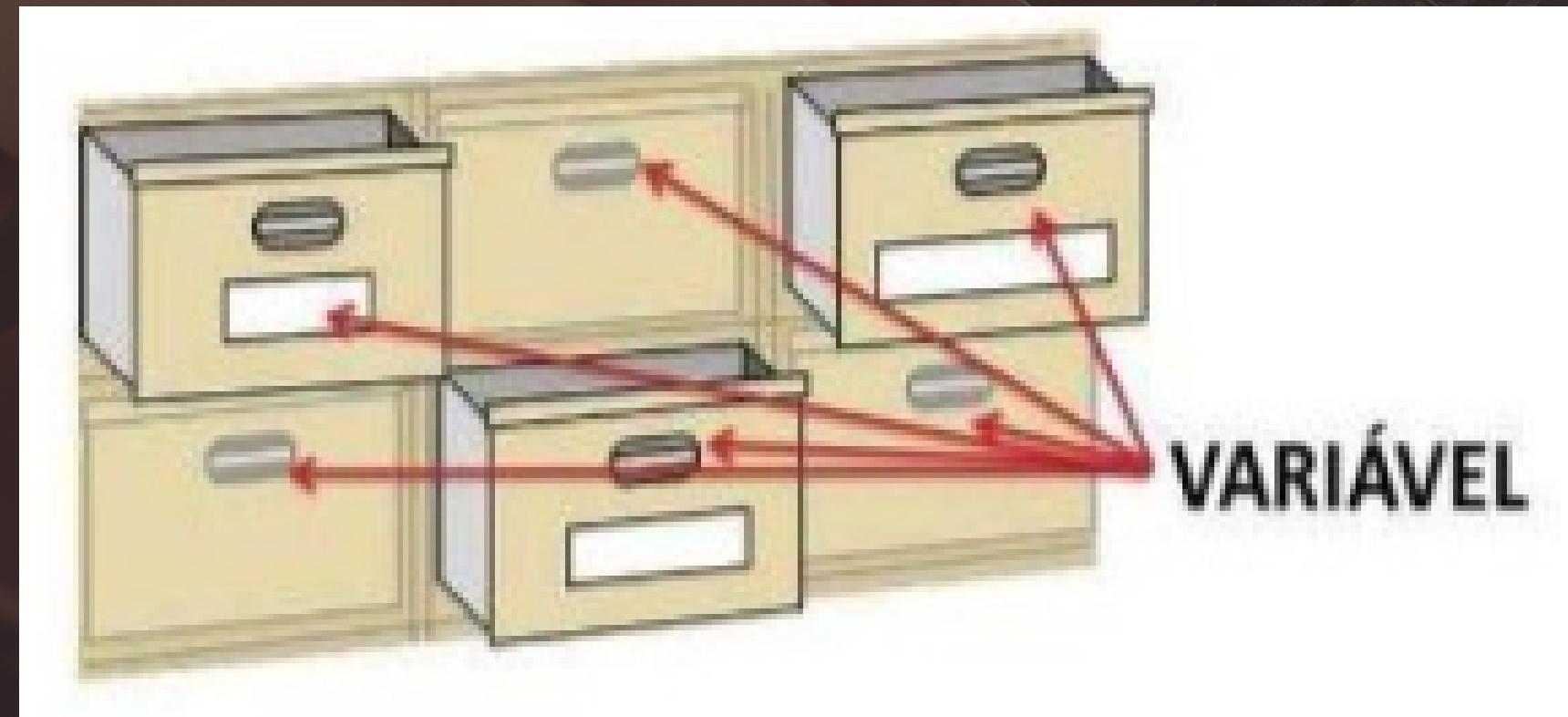
ALGORITMO - conjunto das regras e procedimentos lógicos perfeitamente definidos que levam à solução de um problema

```
algoritmo "numeros"
var
  num_inicial, num_final, i : inteiro
início
  escreva ("Digite o número inicial: ")
  leia (num_inicial)
  escreva ("Digite o número final: ")
  leia (num_final)
  i <- num_inicial
  repita
    escreval (i)
    i <- i + 1
  ate (i > num_final)
  escreva ("fim")
fimalgoritmo
```



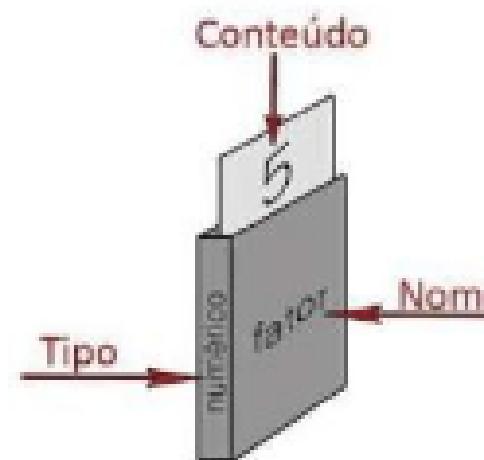
VARIÁVEL

Uma variável pode ser definida como uma caixa ou gaveta com um nome associado à mesma e que em um determinado momento terá um conteúdo a ser guardado. Este conteúdo poderá ser do tipo número, letra, símbolo, entre outros. Na figura a seguir, vemos uma representação para as variáveis de memória.



Os dados que os programas utilizam são armazenados em variáveis e, como já comentamos, uma **variável** é associada a uma posição da **memória do tipo RAM***. Variáveis podem armazenar dados de vários tipos: numéricos, strings (texto), booleanos (verdadeiro ou falso, entre outros).

O nome de uma variável é utilizado para sua identificação e posterior uso em um programa.



<TIPO_VARIAVEL> nomeVariavel;

Exemplo:

```
int anoNascimento;  
float alturaJogador;  
char letra;
```

Variável

- Uma variável é uma estrutura que permite armazenar dados na memória durante a execução do programa, para processamento de informações.
- Todas as variáveis devem ser declaradas antes que possam ser usadas. Declarar uma variável significa criá-la em algum ponto do programa
- A linguagem Java é **fortemente tipada**. Isso significa que cada variável obrigatoriamente deve ter um tipo declarado antes que possa ser utilizada.

Regras para criação de identificadores

Para criar um identificador (nome da variável) em Java, precisamos seguir algumas regras, listadas a seguir:

- Deve conter apenas letras, _ (underline), \$ ou os números de 0 a 9
- Deve obrigatoriamente se iniciar por uma letra (preferencialmente), _ ou \$
- Deve iniciar com uma letra minúscula (boa prática)
- Não pode conter espaços
- Não podemos usar palavras-chave da linguagem
- Além disso, o Java é case sensitive, o que significa que os nomes de variáveis diferenciam maiúsculas de minúsculas.

Exemplos de nomes de variáveis

- Válidos:

- nomeCliente
- telefone_1
- preço\$
- produtoAdquirido

- Inválidos:

- 1Telefone
- Nome Cliente
- #Preço

Declaração de Variáveis

- Vejamos como proceder para realizar a declaração de variáveis em Java.

Sintaxe:

tipo identificador [= valor];

Onde tipo é o tipo de dado que a variável irá armazenar, identificador é seu nome, e valor é o valor inicial atribuído à variável, o qual é opcional (denotado pelos colchetes, que não devem ser digitados na declaração).

Tipos das Variáveis

	TIPO	MEMÓRIA FAIXA	
Lógico	boolean	1 bit	true ou false
Texto	char	2 bytes	\u0000 a \uFFFF
	String	variável	\u0000 a \uFFFF em cada localização
Integral	byte	1 byte	-128 a 127
	short	2 bytes	-32,768 a 32,767
	int	4 bytes	-2,147,483,648 a 2,147,483,647
	long	8 bytes	-9,223,372,036,854,775,808L a 9,223,372,036,854,775,807L
Ponto flutuante	float	4 bytes	aproximadamente +/-3.40282347E+38F (7 dígitos decimais significantes)
	double	8 bytes	aproximadamente +/- 1.79769313486231570E+308 (15 dígitos decimais significantes)

Atribuição às Variáveis

- A ATRIBUIÇÃO É REALIZADA COM O OPERADOR ‘=’
- INICIALIZAÇÃO PADRÃO JAVA
- variáveis numéricas com 0;
- variáveis booleanas com false;
- outras variáveis com null.

Exemplo de Declaração de Variáveis

```
7 public static void main(String[] args) {  
8     boolean praia = true;  
9     byte a;  
10    char letra = 'G';  
11    int valor = 4500;  
12    double x = 3.14;  
13    a = 31;  
14    System.out.printf("Valor de VouF (boolean): %b\n", praia);  
15    System.out.printf("Valor de a (byte): %d\n", a);  
16    System.out.printf("Valor de letra (char): %c\n", letra);  
17    System.out.printf("Valor de valor (int): %d\n", valor);  
18    System.out.printf("Valor de x (double): %.2f\n", x);  
19}  
20 }
```

Exercício

- Crie um programa que armazene em variáveis os seguintes dados:
 - Seu nome;
 - Seu endereço;
 - Sua idade;
 - Seu estado civil, representado por uma letra;
 - Sua altura (armazenada como tipo float).
- Essas informações devem ser exibidas no console como um pequeno relatório.

Constante

- Uma **constante** é um tipo especial de variável cujo valor, uma vez definido no código, não pode mais ser alterado durante a execução do programa.
- Declaramos uma constante em Java usando a palavra-chave final.

```
public static final int TAXA = 20;
```

```
public static final float TAM_MIN = 20.50;
```

Constante

- **public** significa que as constantes estarão disponíveis (serão acessíveis) em todo o código do projeto.
- **static** indica que somente existirá uma cópia da constante compartilhada entre todas as instâncias de classe (o valor da constante será o mesmo não importa onde e quantas vezes apareça no código).
- A palavra **final** significa que o valor atribuído não poderá ser alterado após a inicialização do elemento – ou seja, se torna um valor constante.

```
public static final int TAXA = 20;
```

```
public static final float TAM_MIN = 20.50;
```

Regras Para Nomear Constantes

- Para nomear uma constante em Java, precisamos seguir algumas regras, listadas a seguir:
- Deve conter apenas letras, _ (underline), \$ ou os números de 0 a 9
- Deve obrigatoriamente se iniciar por uma letra, _ ou \$
- Não podemos usar palavras-chave da linguagem
- O nome deve ser único dentro de um escopo
- Boa prática: Declarar sempre uma constante usando apenas letras maiúsculas, e em caso de palavras compostas, separá-las com um underline (_)

Exemplos de Nomes de Constantes

- Válidos:
- NOME_CLIENTE
- TELEFONE_1
- PRECO_\$
- Inválidos:
- 1Telefone
- Nome Cliente
- \$PRECO

Código Exemplo

```
1 package constante;
2
3 public class ExemploConstante {
4
5     public static final double LARGURA = 10.0;
6
7     public static void main(String[] args) {
8         double compr = 25.0;
9         double res = calculaArea(LARGURA, compr);
10        System.out.println("A área é: " + res);
11    }
12    private static double calculaArea(double largura, double comprimento) {
13        return largura * comprimento;
14    }
15
16 }
```

Código Exemplo

- Em nosso código de exemplo declaramos uma constante de nome LARGURA, do tipo double, que conterá o valor 10.0. Esse valor não poderá ser alterado durante a execução do programa.
- Portanto, ao invocarmos o método calculaArea() devemos informar o valor da largura passando a constante LARGURA e o valor de uma variável chamada de compr, que armazena um valor de comprimento. Assim, a área sempre será calculada com a mesma largura, variando apenas o comprimento usado no cálculo.

```
1 package constante;
2
3 public class ExemploConstante {
4
5     public static final double LARGURA = 10.0;
6
7     public static void main(String[] args) {
8         double compr = 25.0;
9         double res = calculaArea(LARGURA, compr);
10        System.out.println("A área é: " + res);
11    }
12    private static double calculaArea(double largura, double comprimento) {
13        return largura * comprimento;
14    }
15
16 }
```

Exercícios

1. Escreva um programa que, com base em uma temperatura em graus celsius (considerar o valor 5.5) , a converta e exiba em Kelvin (K), Réaumur (Re), Rankine (Ra) e Fahrenheit (F), seguindo as fórmulas:

$$F = C * 1.8 + 32;$$

$$K = C + 273.15;$$

$$Re = C * 0.8;$$

$$Ra = C * 1.8 + 32 + 459.67$$

Exercícios

2. Crie um programa que calcule a média de salários de uma empresa com base nos dados abaixo:

- Funcionário 1: R\$ 3453,21
- Funcionário 2: R\$ 3498,43
- Funcionário 3: R\$ 7902,09
- Funcionário 4: R\$ 12932,00

Exercícios

3 . Calcular e apresentar o valor do volume de uma lata de óleo, utilizando fórmula: $V = 3.14 * R * R * A$, em que as variáveis: V, R e A representam respectivamente o volume, o raio e a altura.

Utilizar R= 3.2 e A= 4.9

4. Escreva um programa para determinar a quantidade de cavalos necessários para se levantar uma massa de m quilogramas a uma altura de h metros em t segundos. Considere cavalos = $(m * h / t) / 745,6999$