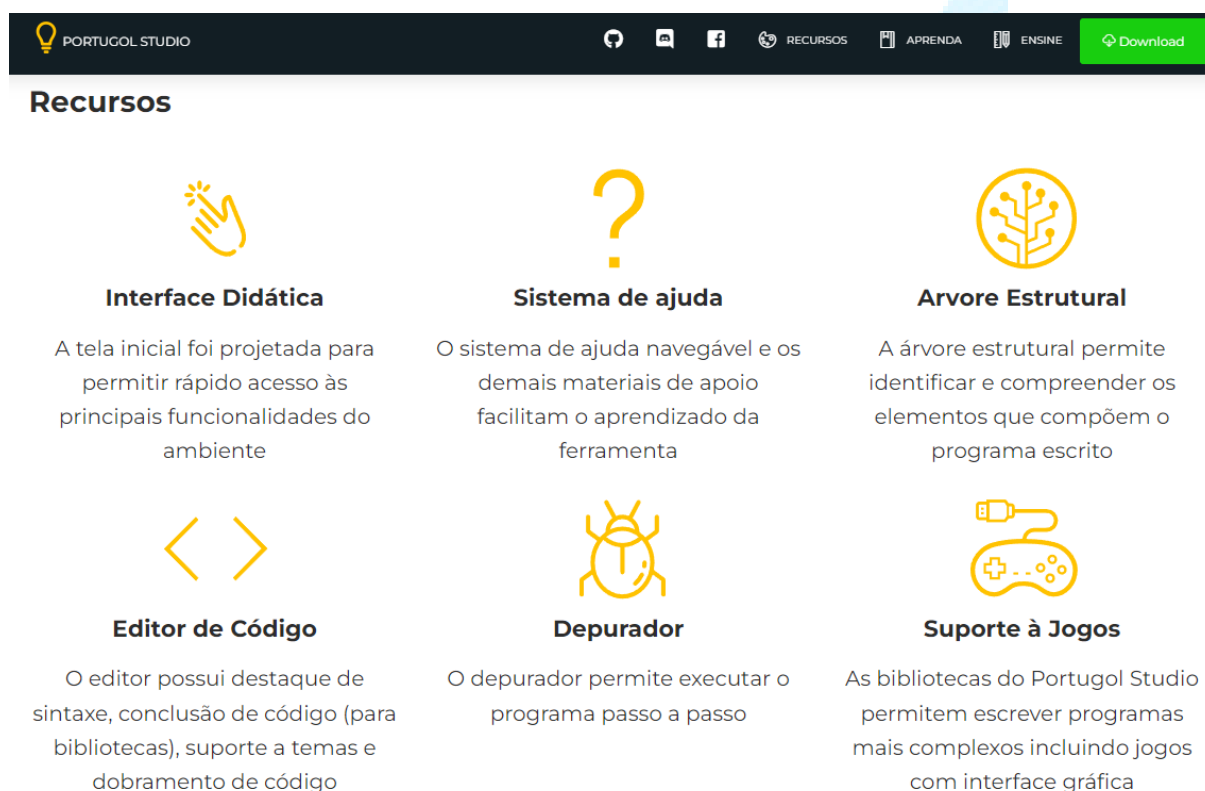


## Portugol Studio

O Portugol Studio é um software para auxiliar o aprendizado de programação de computadores, pois ele é muito semelhante com softwares utilizados no desenvolvimento de programas reais. Sua principal vantagem é que seus comandos são escritos na língua portuguesa e consequentemente os iniciantes em programação também entendem os códigos com mais facilidade.

A Figura 1 descreve vários recursos que estão disponíveis atualmente no Portugol Studio.

Figura 1: Recursos do Portugol Studio



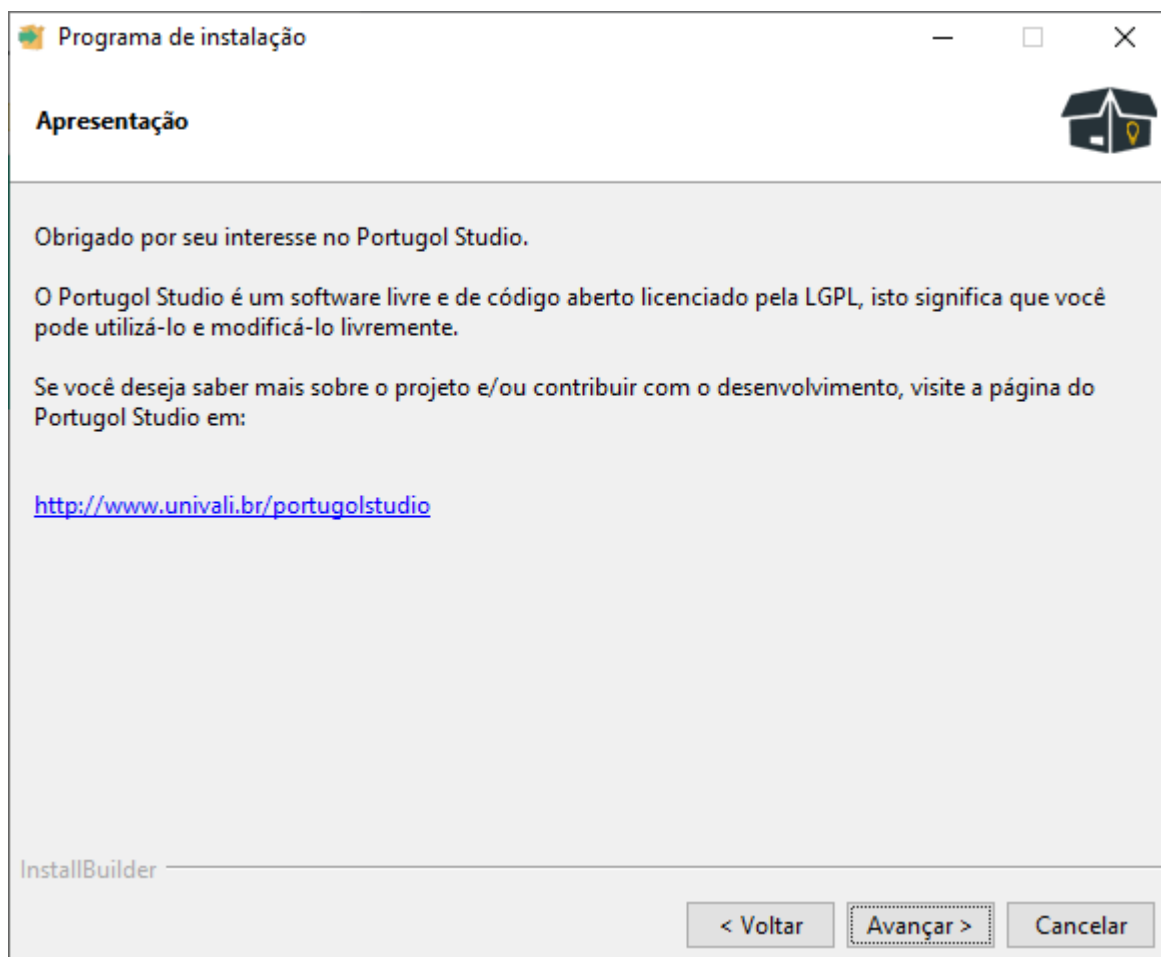
Fonte: <http://lite.acad.univali.br/portugol/>, 2022

O link para fazer o download do Portugol Studio é <http://lite.acad.univali.br/portugol/>. Ele é compatível com os sistemas Windows, Linux e MAC OS.

No início da instalação, a etapa Apresentação (Figura 2) descreve que o software pode ser utilizado e modificado livremente, já que é Portugol

Studio é licenciado pela LGPL (GNU Lesser General Public Licence), uma licença de software livre.

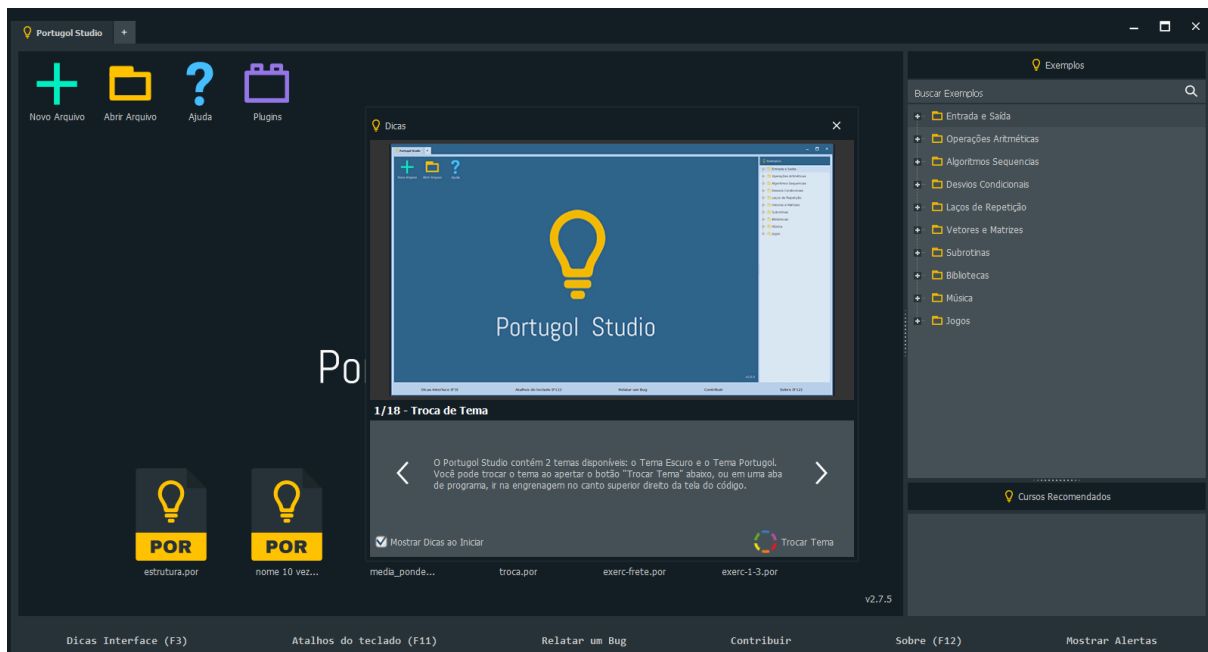
Figura 2: Instalação do Portugol Studio (Etapa Apresentação)



Fonte: portugol-studio-2.7.5-windows.exe, 2022

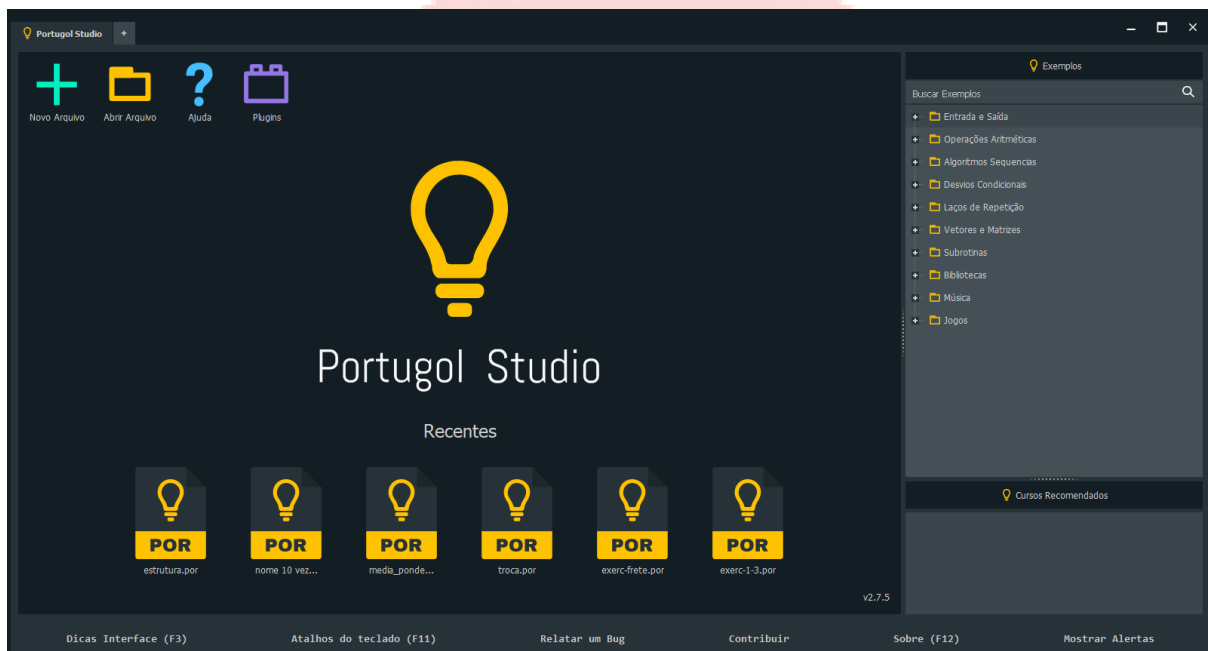
Ao abrir o Portugol Studio pela primeira vez, a tela inicial é apresentada ao fundo da janela Dicas (Figura 3), que poderá ter a opção com opções "Mostrar Dicas ao Iniciar" desmarcada para que não aparece novamente quando o software é executado. Fechando Dicas, a tela apresenta opções para novos arquivos, exemplos e arquivos recentes (Figura 4).

Figura 43: Tela inicial do Portugol Studio com Dicas




Fonte: portugol-studio.jar, 2022

Figura 4: Tela inicial do Portugol Studio



Fonte: portugol-studio.jar, 2022

Clicando na opção “Novo Arquivo”, uma nova aba é criada denominada “Sem título1” e é neste local que o Portugol Studio habilita para programar, inserindo a estrutura do programa que contém “programa” e “funcao inicio()”, acrescentando também o comando “escreva” com o texto “Olá

Mundo". Criando desta forma, um programa simples que já pode ser executado quando clicar em  (Executa o programa até o próximo ponto de parada).

O programa é executado na Console, que fica localizada logo abaixo do código do programa. Para este programa, é exibido o texto do escreva nesta área (Figura 5).

Figura 5: Primeiro programa apresentado pelo Portugol Studio



Fonte: portugol-studio.jar, 2022

Em Portugol Studio, como em qualquer linguagem de programação permite a inclusão de comentários que são ignorados pelo compilador. É uma forma de escrever textos, explicações em qualquer parte do programa.

// → comentário de linha  
/\* ... \*/ → comentário de bloco (mais de uma linha)

A Figura 6 demonstra a estrutura básica de um programa no Portugol Studio com as explicações por meio de comentários.

Figura 6: Estrutura básica de um programa no Portugol Studio


```
//Comando programa é obrigatório
programa
{
    // inclusão de bibliotecas


    // declaração de variáveis globais
    // declaração de constantes globais



    // declaração de funções

    // função principal e obrigatória
    funcao inicio()
    {
        /*
        declaração de variáveis locais
        declaração de constantes locais
        estruturas de controle e expressões
        */
    }
}
```


Fonte: Elaborado pela autora


Ao invés de executar o programa diretamente, há a opção dele ser executado no modo depuração. se clicar em  (Executa o programa passo a passo) para se fazer testes, pois é possível verificar cada linha que é executada e o programa só avança para a linha seguinte quando clicar novamente neste mesmo ícone.

Para interromper a execução, em quaisquer um dos modos de execução, clica-se em  (Interrompe a execução / depuração do programa atual).

Para salvar o programa, clica-se em  (Salva o programa atual no computador em uma pasta escolhida pelo usuário) ou em  (Salva uma nova cópia do programa atual no computador em uma pasta escolhida pelo

usuário). Os arquivos dos programas feitos no Portugol Studio são salvos com a extensão “.por”.

Para abrir um arquivo já salvo, basta clicar em  (Abre um arquivo .por) que apresenta a caixa de diálogo “Abrir” para selecionar o arquivo desejado.

Ainda nesta tela, há o ícone  (Abre a ajuda com sintaxe e bibliotecas).

Estes dois últimos também estão disponíveis na tela inicial do Portugol Studio.

## VOCÊ SABIA?

Há um ambiente online denominado de Portugol WebStudio que foi construído para utilizar os comandos do Portugol Studio diretamente no navegador web.

Pode ser acessado em: <https://portugol-webstudio.cubos.io/>.

## Bibliotecas

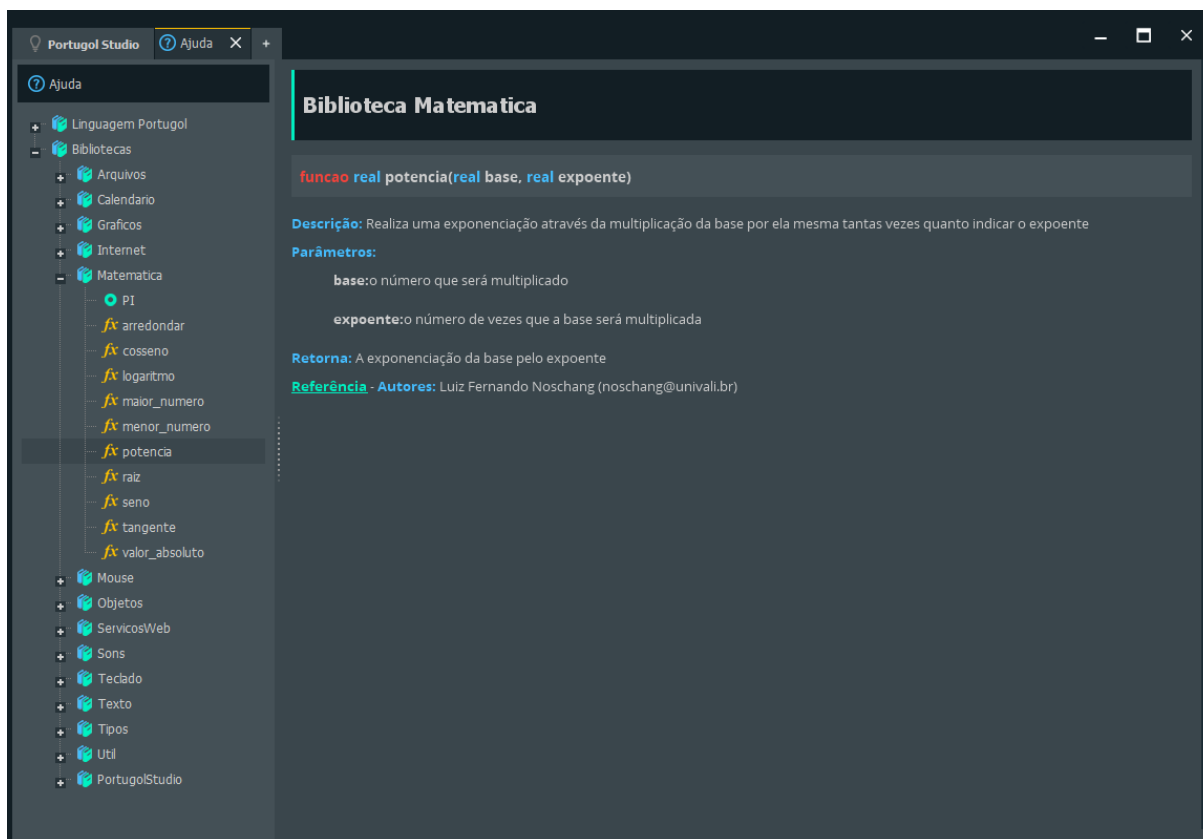
O Portugol Studio possui funções pré-existentes que são encontradas em Bibliotecas, quando se clica no ícone Ajuda.

Cada biblioteca possui as funções que atendem aquele tipo de requisito.

O exemplo da Figura 7 demonstra a aba Ajuda com a árvore “Matemática” expandida e nela selecionada a função “potencia”. Ao selecionar uma função é apresentado sua sintaxe para utilizá-la corretamente, sua descrição, o significado de seus parâmetros, caso possua, o que retorna e os autores responsáveis pela sua criação.



Figura 7: Bibliotecas do Portugol Studio



Fonte: portugol-studio.jar, 2022

Para utilizar uma função da biblioteca, primeiro deve-se importar a biblioteca a qual essa função pertence através do comando **"inclua biblioteca nome-da-biblioteca"** (Figura 8).

Figura 8: Inclusão de Biblioteca no programa

```
programa
{
    inclua biblioteca Matematica

    funcao inicio()
    {

    }
}
```

Fonte: Elaborado pela autora

Ao utilizar a função, deve-se iniciar pelo nome da biblioteca, acrescentar um ponto e em seguida a função desejada com parênteses e

seus respectivos parâmetros, se forem necessários (Figura 9).

Figura 9: Uso de Função pré-existente

```
programa
{
    inclui biblioteca Matematica

    funcao inicio()
    {
        escreva(Matematica.potencia(2.0, 3.0))
    }
}
```

Fonte: Elaborado pela autora

## VOCÊ SABIA?

É possível atribuir um apelido para a biblioteca no programa, utilizando "-->". Assim, ao descrever o seu nome para utilizar com suas funções, a escrita do código poderá ficar menor (Figura 10).

Figura 10: Definindo apelina na inclusão de biblioteca

```
programa
{
    inclui biblioteca Matematica --> m

    funcao inicio()
    {
        escreva(m.potencia(2.0, 3.0))
    }
}
```

Fonte: Elaborado pela autora

Por se tratar de um software de código aberto, muitas bibliotecas ainda podem surgir, de acordo com a contribuição da comunidade.

## Tipos de Dados

Tipos de dados são definidos para armazenar valores no



computador. A escolha correta do tipo de dado em um programa otimiza a utilização da memória. Quando a linguagem de programação não oferece essa especificação, os tipos de dados são identificados dinamicamente.

O Portugol Studio possui alguns tipos básicos, utilizados para armazenar os principais tipos de conteúdos:

- Tipo Cadeia: Para um texto ou muitos caracteres.

**cadeia** nome\_da\_cadeia

- Tipo Caracter: Para um único caracter qualquer.

**caracter** nome\_do\_caracter

- Tipo Inteiro: Para valores inteiros.

**inteiro** nome\_do\_inteiro

- Tipo Real: Para valores com ponto flutuante.

**real** nome\_do\_real

- Tipo Lógico: Para informações do tipo verdadeiro e falso.

**logico** nome\_do\_lógico

- Tipo Vazio: Utilizado na criação de funções quando não há retorno de valores após o seu término.

**funcao vazio** nome\_da\_função()

A Figura 11 demonstra exemplo da definição de cada um dos tipos presentes no Portugol Studio.

Figura 11: Definição de Tipos de Dados no Portugol Studio

```
programa
{
    funcao inicio()
    {
        cadeia mensagem

        caracter letra

        inteiro idade

        real preco

        logico teste
    }

    funcao vaziao traco()
    {
        escreva("-----")
    }
}
```

Fonte: Elaborado pela autora

## Declaração de Variáveis

Variável é um espaço reservado na memória RAM (*Random Access Memory*) do computador que não possui valor fixo, ou seja, pode variar durante a execução do programa. Ao declarar uma variável, o Portugol Studio reserva um espaço na memória para armazenar o valor que ela conterá de acordo com seu tipo definido.

Quando o usuário informa algum valor, esse conteúdo é armazenado em uma variável para ser utilizada em qualquer parte do código.

Para declarar variáveis no Portugol Studio, inicia-se definindo o tipo de dado seguido do seu nome:

**tipo\_de\_dado** nome\_variável.

Também é possível declarar mais de uma variável de mesmo tipo em uma única linha, separando-as com vírgula:

**tipo\_de\_dado** nome\_variável1, nome\_variável2

A atribuição de um valor inicial para a variável pode ser feita diretamente na sua declaração acrescentando o sinal de igual e o valor

desejado:

**tipo\_de\_dado** nome\_variável = valor\_inicial

A Figura 12 apresenta exemplos de declarações de variáveis e a atribuição de valores para cada uma delas através do sinal de igual "=" após o seu nome.

Figura 12: Declaração de variáveis no Portugol Studio

```
programa
{
    funcao inicio()
    {
        cadeia aviso // Declaração de uma variável

        real nota1, nota2 // Declaração de duas variáveis

        inteiro X = 10 // Declaração de uma variável com valor inicial

        // Atribuindo valores para as variáveis
        aviso = "Estude Portugol Studio"
        nota1 = 7.5
        nota2 = 10.0
    }
}
```

Fonte: Elaborado pela autora

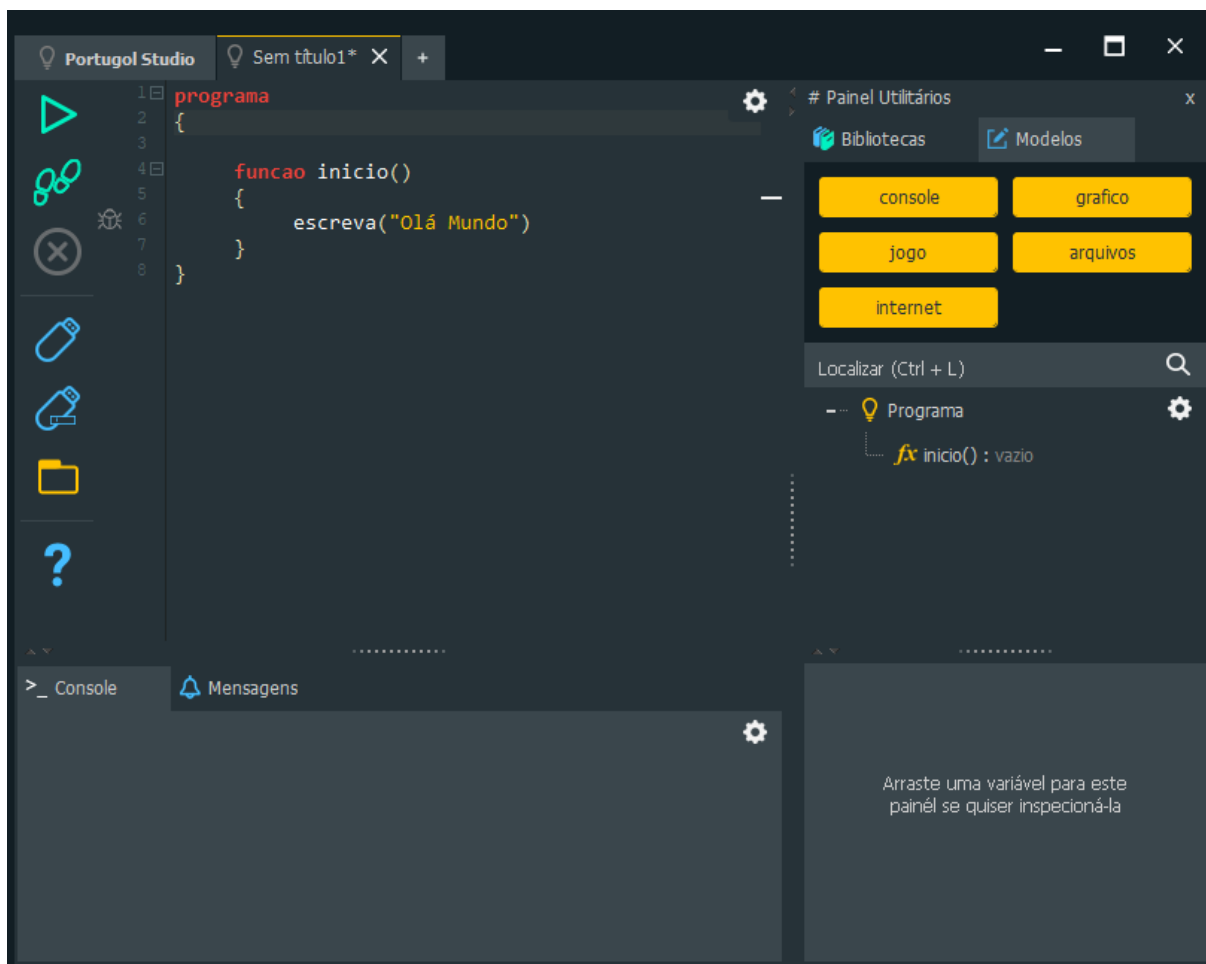
As variáveis só podem receber valores do mesmo tipo que elas foram declaradas.

Há regras para nomear variáveis:

1. Formado apenas por letras, números e sublinhado ("\_")
2. Deve começar com uma letra
3. Não pode ser igual a palavras reservadas do Portugol Studio, ou seja, palavras que já fazem parte da linguagem, como os comandos.

No ambiente do Portugol Studio também há uma área para inspecionar variáveis, como pode ser observado na Figura 13 no canto inferior direito.

Figura 13: Bibliotecas do Portugol Studio



Fonte: portugol-studio.jar, 2022

Verificar o conteúdo das variáveis é importante quando se executa o programa no modo passo a passo, pois cada vez que a variável é modificada, seu novo valor pode ser visto nesta área do ambiente. Porém, atente que a alteração é efetada somente após a saída da linha onde o programa se encontra no modo debug, pois somente após seguir para a próxima linha é que o comando foi realmente executado de acordo com sua finalidade.

## Declaração de Constantes

Constantes são utilizadas quando um tipo reservado na memória não pode ter seu valor alterado durante a execução do programa.

Para isso adiciona-se a palavra “**const**” ao tipo de dado, seguido pelo

seu nome e o valor atribuído a ela:

**const tipo\_de\_dado** NOME\_CONSTANTE = valor

A Figura 14 demonstra um exemplo de declaração de constante. É aconselhável que o nome da constante seja escrita toda em letras maiúsculas como boa prática de programação.

Figura 14: Declaração de constante no Portugol Studio

```
programa
{
    funcao inicio()
    {
        const real PI = 3.14
    }
}
```

Fonte: Elaborado pela autora

As regras para nomear as constantes são as mesmas para os nomes das variáveis.

## Operações Aritméticas

Os operadores aritméticos são utilizados nas linguagens de programação para efetuar cálculos.

O Quadro 1 apresenta os símbolos do Portugol Studio para cada uma das operações.

Quadro 1: Operadores Aritméticos

Símbolo	Operação
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Módulo (resto da divisão inteira)

Fonte: Elaborado pela autora

As operações podem ser feitas diretamente em um comando de tela, o qual escreve a resposta e/ou atribuídas em variáveis. Também é permitido na inicialização da declaração de variáveis (Figura 15).

Figura 15: Exemplos de operações aritméticas

```
programa
{
    funcao inicio()
    {
        //Operação atribuída na declaração da variável
        inteiro resp = 1 + 2

        // Operações atribuídas em variáveis
        inteiro x, y, soma, sub, mult, div, resto
        x = 10
        y = 5
        soma = x + y
        sub = x - y
        mult = x * y
        div = x / y
        resto = x % y

        // Operação escrita na tela
        escreva(1+2)
    }
}
```

Fonte: Elaborado pela autora

Como já se conhece da matemática, quando há várias operações para serem realizadas juntas, a multiplicação, divisão e módulo têm prioridade sobre adição e subtração. Para modificar a ordem em que as operações são efetuadas indica-se o uso de parênteses (Figura 16).

Figura 16: Alteração de prioridades em operações matemáticas

```
programa
{
    funcao inicio()
    {
        inteiro x, y, z, resp
        x = 10
        y = 5
        z = 2

        resp = x + y * 2 // resp = 20

        resp = (x + y) * 2 // resp = 30
    }
}
```

Fonte: Elaborado pela autora

O operador “+” também é utilizado para concatenar valores do tipo cadeia e caracter (Figura 17).

Figura 17: Concatenação de frases

```
programa
{
    funcao inicio()
    {
        cadeia saudacao, cumprimento, mensagem

        saudacao = "Olá."
        cumprimento = "Como vai você?"

        mensagem = saudacao + cumprimento
        // "Olá.Como vai você?" é atribuído em mensagem
    }
}
```

Fonte: Elaborado pela autora

## Operações Relacionais

Operadores relacionais são utilizados para comparar valores resultando em um valor booleano, ou seja, um tipo de dado lógico que pode ser apenas verdadeiro ou falso.

O Quadro 2 apresenta os símbolos para operadores relacionais.

Quadro 2: Operadores Relacionais

Símbolo	Operação
<	Menor
>	Maior
<=	Menor ou igual
>=	Maior ou igual
==	Igual
!=	Diferente

Fonte: Elaborado pela autora

A Figura 18 apresenta exemplo de operadores relacionais, onde o resultado de cada uma das operações é atribuído nas variáveis lógicas.

Figura 18: Exemplos de operações relacionais

```
programa
{
    funcao inicio()
    {
        inteiro a, b
        logico menor, maior, igual, dif

        a = 1
        b = 2

        menor = a < b // menor = verdadeiro
        maior = a > b // maior = falso
        igual = a == b // igual = falso
        dif = a != b // dif = verdadeiro
    }
}
```

Fonte: Elaborado pela autora

É possível executar operadores aritméticos e relacionais juntos. Quando isso ocorre, a prioridade é executar primeiro a operação aritmética e depois a operação relacional.

A Figura 19 demonstra como o resultado das operações relacionais e aritméticas juntas são atribuídas às variáveis lógicas.

Figura 19: Exemplos de operações aritméticas e relacionais

```
programa
{
    funcao inicio()
    {
        inteiro a, b
        logico result1, result2

        a = 1
        b = 2

        result1 = a <= a + b
        // a <= 3 --> 1 <= 3 --> result1 = verdadeiro

        result2 = b >= b - a
        // b >= 1 --> 2 >= 1 --> result2 = verdadeiro
    }
}
```

Fonte: Elaborado pela autora



## Operações Lógicas

Em computação, as operações lógicas são utilizadas quando realiza operações com valores booleanos. Portanto, podem ser combinadas com operações relacionais. Para isso, o Portugol Studio possui os principais operadores lógicos, os quais têm prioridades entre eles para execução, de acordo com o Quadro 3.

Quadro 3: Operadores Lógicos

Operação	Prioridade
nao	1º
e	2º
ou	3º

Fonte: Elaborado pela autora

O resultado com o operador **"ou"** é verdadeiro quando pelo menos uma das condições for verdadeira, ou seja, se dentre várias condições uma for verdadeira a resposta sempre será verdadeira. O Quadro 4 demonstra a tabela verdade com o comportamento deste operador.

Quadro 4: Tabela verdade para operador **"ou"**

Operação 1	Operação 2	Resultado
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	verdadeiro
falso	verdadeiro	verdadeiro
falso	falso	falso

Fonte: Elaborado pela autora

A Figura 20 demonstra o uso do operador **"ou"** no Portugol Studio.

Figura 20: Operador “ou”

```
programa
{
    funcao inicio()
    {
        inteiro a=1, b=2, c=3, d=4
        logico r

        r = a < b ou c < d
        // r = verdadeiro ou verdadeiro
        // r = verdadeiro

        r = falso ou d <= c
        // r = falso ou falso
        // r = falso

        r = a+b == d ou c != a-b // r = 3 == 4 ou 3 != -1
        // r = falso ou verdadeiro
        // r = verdadeiro
    }
}
```

Fonte: Elaborado pela autora

Entretanto há situações que o resultado deverá ser verdadeiro se todas as condições forem verdadeiras. Neste caso, utiliza-se o operador “e”. Seu comportamento na tabela verdade é apresentado no Quadro 5.

Quadro 5: Tabela verdade para operador “e”

Operação 1	Operação 2	Resultado
verdadeiro	verdadeiro	verdadeiro
verdadeiro	falso	falso
falso	verdadeiro	falso
falso	falso	falso

Fonte: Elaborado pela autora

Seu uso no Portugol Studio é apresentado na Figura 21.

Figura 21: Operador “e”

```
programa
{
    funcao inicio()
    {
        inteiro a=1, b=2, c=3, d=4
        logico r

        r = a < b e c > d
        // r = verdadeiro e falso
        // r = falso

        r = verdadeiro e d >= c
        // r = verdadeiro e verdadeiro
        // r = verdadeiro

        r = a+b == d e c == a-b // r = 3==4 e 3==1
        // r = falso e falso
        // r = falso
    }
}
```

Fonte: Elaborado pela autora

No caso de necessitar inverter o resultado de uma condição, utiliza-se o operador “**nao**”. O Quadro 6, demonstra o seu comportamento que difere dos anteriores, pois age somente em uma operação.

Quadro 6: Tabela verdade para operador “nao”

Operação	Resultado
verdadeiro	falso
falso	verdadeiro

Fonte: Elaborado pela autora

A Figura 22 apresenta o uso do operador “nao” pelo Portugol Studio. Lembrando que, ao combinar a utilização de operadores lógicos, a prioridade apresentada anteriormente no Quadro 4 é seguida.

Figura 22: Operador “**nao**”

```
programa
{
    funcao inicio()
    {
        inteiro a=1, b=2, c=3, d=4
        logico r

        r = nao(a < b e c > d)
        // r = nao(verdadeiro e falso)
        // r = nao(falso)
        // r = verdadeiro

        r = nao(verdadeiro) ou d >= c // r = falso ou d >= c
        // r = falso ou verdadeiro
        // r = verdadeiro

        r = nao(a+b == d e nao(c == a-b))
        // r = nao(3==4 e nao(3== -1))
        // r = nao(3==4 e nao(falso))
        // r = nao(3==4 e verdadeiro)
        // r = nao(falso e verdadeiro)
        // r = nao(falso)
        // r = verdadeiro
    }
}
```

Fonte: Elaborado pela autora

## Entrada e Saída

A comunicação do usuário com o computador se dá por meio de comandos de entrada (inserção de dados) por meio do teclado, mouse ou qualquer outro dispositivo para isso e saída (obtenção de dados), normalmente pela visualização do resultado na tela do computador.

**leia:** No Portugol Studio há o comando “**leia**(variável\_1,variável\_n)”, responsável por capturar dados via teclado, atribuindo diretamente o conteúdo digitado pelo usuário na variável. Portanto as variáveis devem ser declaradas anteriormente e esse comando aceita mais de uma entrada, separadas por vírgula (Figura 23).

Figura 23: Exemplo do comando “leia”

```
programa
{
    funcao inicio()
    {
        inteiro numero
        cadeia texto
        caracter letra

        leia(numero)
        leia(texto, letra)
    }
}
```

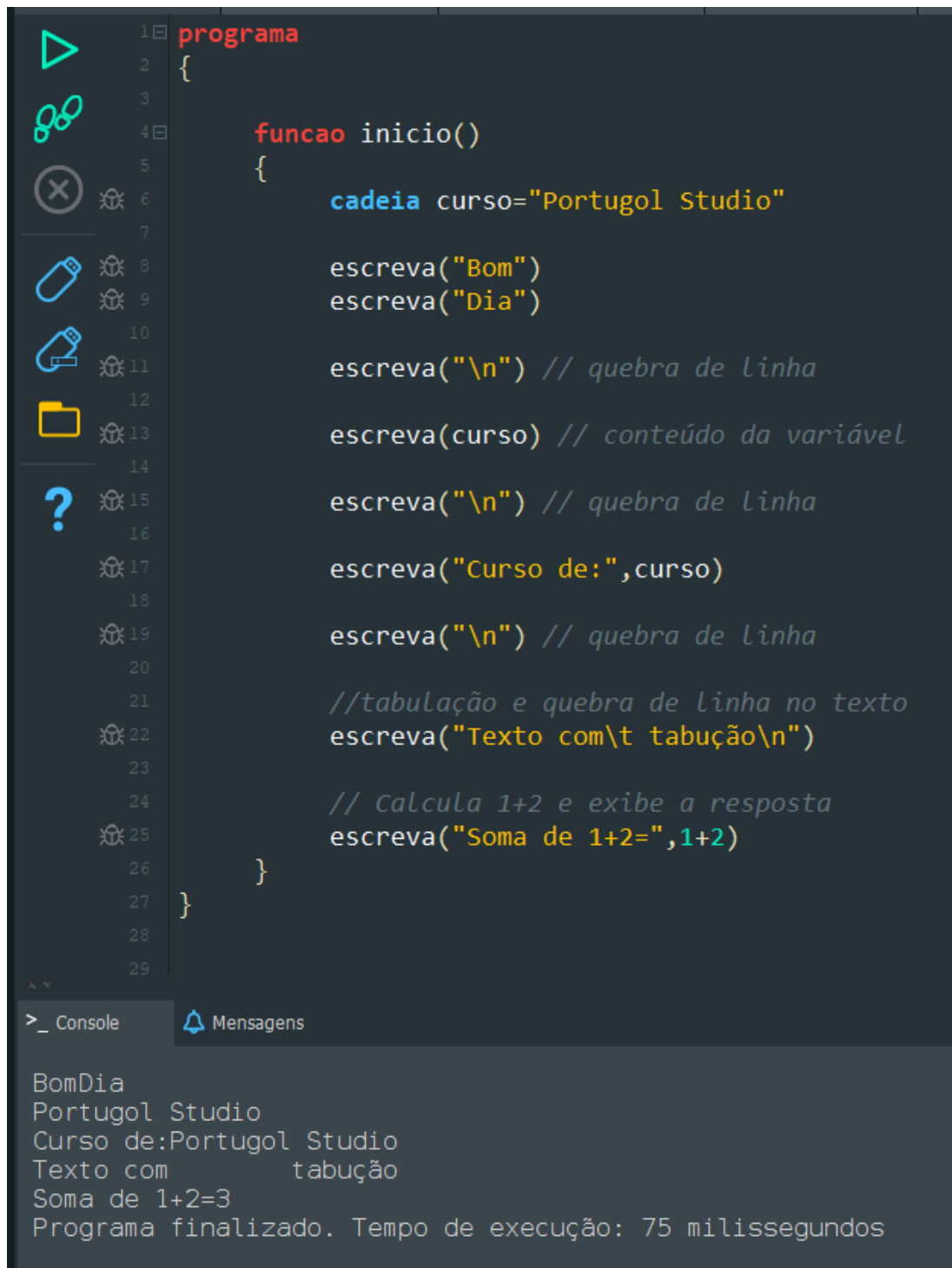
Fonte: Elaborado pela autora

**escreva:** Para a saída de dados na tela do computador (Console do ambiente), o Portugol Studio possui o comando “**escreva**(texto\_ou\_variável\_1, texto\_ou\_variável\_n)”, o qual permite que sejam exibidos textos fixos (devem ser escritos entre aspas) e/ou variáveis.

Para ajudar a organizar os dados na tela, ainda há dois recursos:

- “\n”: insere uma nova linha
- “\t”: insere espaços maiores entre os dados

A Figura 24 demonstra exemplos do uso desse comando, bem com a exibição do resultado na Console do ambiente do Portugol Studio. Observe que o resultado dos dois primeiros comandos “**escreva**” estão juntos, pois apesar de possuírem os textos em comandos separados, não foram utilizados nenhum dos recursos apresentados (“\n” ou “\t”). Já nas próximas saídas, o conteúdo está organizado com o uso de quebra de linha e tabulação. Operações também podem ser realizadas dentro do comando, que exibe os resultados diretamente na tela.

Figura 24: Exemplo do comando “**escreva**”

```
1 programa
2 {
3
4     funcao inicio()
5     {
6         cadeia curso="Portugol Studio"
7
8         escreva("Bom")
9         escreva("Dia")
10
11        escreva("\n") // quebra de linha
12
13        escreva(curso) // conteúdo da variável
14
15        escreva("\n") // quebra de linha
16
17        escreva("Curso de:",curso)
18
19        escreva("\n") // quebra de linha
20
21        //tabulação e quebra de linha no texto
22        escreva("Texto com\t tabução\n")
23
24        // Calcula 1+2 e exibe a resposta
25        escreva("Soma de 1+2=",1+2)
26    }
27 }
28
29
```

> \_ Console    Mensagens

BomDia  
Portugol Studio  
Curso de:Portugol Studio  
Texto com            tabução  
Soma de 1+2=3  
Programa finalizado. Tempo de execução: 75 milissegundos

Fonte: Elaborado pela autora

**limpa:** Durante a execução do programa, a exibição no Console pode ficar com muitos dados desnecessários que podem dificultar o entendimento das informações. Para colaborar neste sentido, o Portugol Studio oferece o comando “**limpa()**”, que é responsável

por apagar todo o conteúdo do Console quando é executado. Pode ser uma ideia solicitar os dados, limpar a tela e depois apresentar somente os resultados, conforme o exemplo da Figura 25.

Figura 25: Exemplo do comando “limpa”

```
programa
{
    funcao inicio()
    {
        cadeia nome

        //imprime a frase "Qual é o seu nome?"
        escreva("Qual é o seu nome ?\n")

        //Detecta o que o usuario escreveu na tela
        leia(nome)

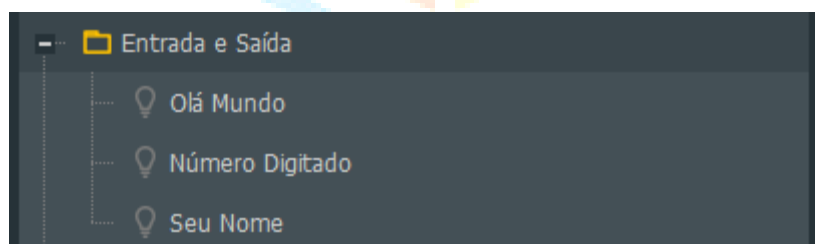
        //Limpa tudo que estava escrito no console
        limpa()

        //Escreve resposta
        escreva("Olá "+nome)
    }
}
```

Fonte: portugol-studio.jar, 2022

Na tela inicial do Portugol Studio ainda é possível verificar exemplos utilizando os comandos de Entrada e Saída (Figura 26).

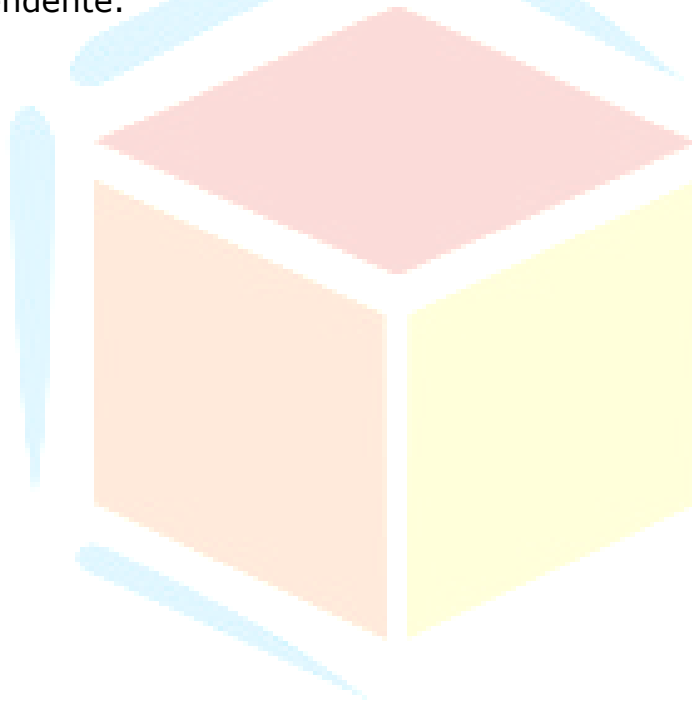
Figura 26: Exemplos programas de Entrada e Saída do Portugol Studio



Fonte: portugol-studio.jar, 2022

## PARA PRATICAR

1. Ler o nome e o sobrenome do usuário e escreva sobrenome, nome.
2. Ler 4 notas, calcular a média aritmética e exibir as notas e o resultado da média.
3. Ler a base e altura de um triângulo. Calcular e exibir a área do triângulo sabendo que  $\text{área} = \text{base} * \text{altura} / 2$ .
4. Ler o nome e a idade do usuário. Calcular a idade em meses e dias. Exibir o nome e a idade em anos, meses e dias.
5. Ler a quantidade de horas trabalhadas no mês pelo funcionário, o número de dependentes e o valor que recebe por hora trabalhada. Calcular e exibir o valor total a receber, sabendo que é acrescido 10% por dependente.





## REFERÊNCIAS

ALVES, Gustavo Furtado de Oliveira. **Conheça os Operadores Relacionais!**. Disponível em: <<https://dicasdeprogramacao.com.br/operadores-relacionais/>> Acesso em 16 mar. 2022.

Colibri Team. **Licença LGPL**. Disponível em: <<https://wiki.ncrcolibri.com.br/pages/viewpage.action?pageId=3558608>> . Acesso em 11 mar. 2022.

Cursos NCE a Distância - NCE – UFRJ. **Nomes de variáveis**. Disponível em: <<http://www.nce.ufrj.br/ginape/js/conteudo/variaveis/nomes.htm>>. Acesso em 13 mar. 2022.

DEVMEDIA. **Algoritmos: Entrada e Saída de dados**. Disponível em: <<https://www.devmedia.com.br/algoritmos-entrada-e-saida-de-dados/40748>>. Acesso em 20 mar. 2022.

GATTO, Elaine Cecília. **Tipos de dados para uso em algoritmos**. Embarcados, 2016. Disponível em: <<https://www.embarcados.com.br/tipos-de-dados/>>. Acesso em 12 mar. 2022.

GATTO, Elaine Cecília. **Variáveis e Constantes**. Embarcados, 2016. Disponível em: <<https://www.embarcados.com.br/variaveis-e-constantes/>>. Acesso em 13 mar. 2022.

GOUVEIA, Rosimar. **Lógica Matemática**. Disponível em: <<https://www.todamateria.com.br/logica-matematica/>>. Acesso em 17 mar. 2022.

MORENO, Heliete Martis Castilho. **As operações aritméticas fundamentais**. UFMT em Rede, 2021. Disponível em: <[https://setec.ufmt.br/ri/bitstream/1/85/3/Operacoes\\_aritmeticas\\_2021.pdf](https://setec.ufmt.br/ri/bitstream/1/85/3/Operacoes_aritmeticas_2021.pdf)> Acesso em 15 mar. 2022.

PORTUGOL STUDIO. **Portugol Studio**. Disponível em: <<http://lite.acad.univali.br/portugol/>>. Acesso em 10 mar. 2022.