M05

Fundamentos de Programação

Prof. Romário Nzenguele da Silva

Licenciatura em Informática

2024 - 2025

Sobre o Documento

Material de apoio **gratuito**, desenvolvido exclusivamente para os estudantes do **1º Ano da Licenciatura em Informática**;

O conteúdo, sempre que necessário, será atualizado para refletir as mudanças e melhorias no decorrer do curso. Cada atualização será devidamente versionada, permitindo o fácil acompanhamento das revisões;

Este recurso é um **resumo das aulas**, essencial para apoiar no aprendizado da disciplina.

O **repositório da disciplina** (contendo exercícios, conteúdos detalhados, programa...) será reencaminhado por email;

Sugestões de melhoria podem ser apresentadas pelo email que se encontra no rodapé.

Conteúdo

5. Controlo de Fluxo

5.1. Estruturas condicionais

5.2. Estruturas de repetição

Objectivos

- ✓ Compreender o conceito de controlo de fluxo;
- ✓ Implementar blocos condicionais em algoritmos.
- ✓ Implementar blocos cíclicos em algoritmos.

5. Controlo de fluxo

- Determina a ordem em que as instruções são executadas em um programa.
- Por padrão, a execução é **sequencial** (linha por linha). Porém, podemos implementar blocos condicionais e/ou blocos de repetição.

Suporte: Portugol (Portugol Studio)

5.1. Estruturas Condicionais

- Fundamentais para o desenvolvimento de algoritmos. Permitem a execução de diferentes ações dependendo de determinadas condições.

- Dentre elas, temos:

  - **Se:** executa um bloco de código se uma condição for verdadeira.
  - **Se-senao:** executa um bloco de código se a condição(no **se**) for falsa.
  - **Se-senao se:** permite testar várias possibilidades.
  - **escolha-caso:** utilizada quando várias alternativas são associadas ao valor de uma variável.

Para mais detalhes, ver a documentação :

1º- Clicar em Ajuda;
2º- Clicar em Estruturas de Controle.
3º- Clicar em Desvios Condicionais.

Link: https://portugol.dev/

5.1. Estruturas Condicionais

❑ **Sintaxe:**

Bloco Se

```
se (<condição>){       //se verdadeiro
    <instruções>;    //faça isso
}
```

Bloco Se-senao

```
se (<condição>){          //se verdadeiro
    <instruções>;        //faça isso
} senao {                    //se falso
   <instruções>;       //faça isso
}
```

```
real nota = 7

//Bloco Se
se(nota>=9.5){
      escreva("Aprovado")
}

//Bloco Se-senao
se(nota>=9.5){
      escreva("Aprovado")
}senao{
      escreva("Reprovado")
}
```

5.1. Estruturas Condicionais

❑ **Sintaxe:**

Bloco Se-senao se

```
se (<condição>){          //se verdadeiro
    <instruções>;         //faça isso
} senao se (<condição>){  //senão, testa nova condição. Se verdadeiro
    <instruções>;         //faça isso
} senao {                 //se falso
   <instruções>;          //faça isso
}
```

```
real nota = 7

//Bloco Se-senao se
se(nota>=9.5){
      escreva("Aprovado")
}senao se(nota>=7 e nota<9.5){
      escreva("Recurso")
}senao{
      escreva("Reprovado")
}
```

5.1. Estruturas Condicionais

- **Sintaxe:**

Bloco escolha-caso

```
escolha (<variável>){
    caso <valor>:
        <instruções>;
        pare;
    caso contrario:
        <instruções>;
        pare;
}
```

Nota: O numero de **casos** depende das opções a serem implementadas. Cada case representa uma condição.

```
escolha(dia)
{
  caso 1:
      escreva("Segunda-feira")
      pare
  caso 2:
      escreva("Terça-feira")
      pare
  caso 3:
      escreva("Quarta-feira")
      pare
  caso 4:
      escreva("Quinta-feira")
      pare
  caso 5:
      escreva("Sexta-feira")
      pare
  caso contrario:
      escreva("Número inválido. Inserir números entre 1 e 5.")
}
```

5.2. Estruturas de repetição

- Assim como as condicionais, são fundamentais para o desenvolvimento de algoritmos.
- Permitem a execução de ciclos ou laços de repetição, obedecendo a uma determinada condição.

- O Portugol Studio suporta os três laços principais:

Para mais detalhes, ver a documentação:

1º- Clicar em Ajuda;
2º- Clicar em Estruturas de Controle;
3º- Clicar em Laços de repetição.

Link: https://portugol.dev/

para
(melhor para)

- ✓ Sequências (contadores) que seguem um padrão específico;
- ✓ Percorrer vetores;
- ✓ Quando sabes exatamente a quantidade de iterações necessárias.

enquanto
(melhor quando)

- ✓ A condição de parada depende de um evento externo (entrada do usuário, arquivo, etc);
- ✓ A quantidade de iterações necessárias é desconhecida.

faca-enquanto
(melhor quando)

- ✓ Precisamos garantir que o código execute pelo menos uma vez;

5.2. Estruturas de repetição

- **Sintaxe:**

Laço para

```
para (<inicialização>; <condição>; <incremento/decremento>){
        <instruções>;
}
```

```
escreva("1. Contagem regressiva:\n")
para(inteiro i = 10; i >= 1; i--)
{
    escreva(i, "... ")
}
```

Laço enquanto

```
<inicialização>
enquanto (<condição>){
        <instruções>;
        <incremento/decremento>;
}
```

```
enquanto(palpite != numero)
{
    escreva("Entre 1 e 10, seu palpite: ")
    leia(palpite)

    se(palpite != numero)
    {
        escreva("Errado! Tenta novamente!\n")
    }senao{
        escreva("Parabéns! Você acertou!\n\n")
    }
}
```

5.2. Estruturas de repetição

❑ **Sintaxe:**

Laço faca-enquanto

<inicialização>

faca{

<instruções>;

<incremento/decremento>;

} enquanto (<condição>);

```
faca
{
    escreva("\nMENU:\n")
    escreva("1 - Ver horário\n")
    escreva("2 - Ver data\n")
    escreva("3 - Sair\n")
    escreva("Escolha uma opção: ")
    leia(opcao)

    escolha(opcao)
    {
        caso 1:
            escreva("Agora são 15:30\n")
            pare
        caso 2:
            escreva("Hoje é 16/11/2024\n")
            pare
        caso 3:
            escreva("Saindo do programa...\n")
            pare
        caso contrario:
            escreva("Opção inválida!\n")
    }
}enquanto(opcao != 3)
```

ANEXOS

Caderno de exercícios

Lista de exercícios para consolidar o aprendizado.

Consultar no repositório da turma.

Recomendações

- ✓ [Cadastro] **GitHub**: https://github.com/signup?source=login
- ✓ [Download] **Git**: https://git-scm.com/downloads
- ✓ [Tutorial] **GitHub/Git**: https://docs.github.com/pt/get-started/start-your-journey

...

Sequência -> **M06**