

Programação 2: Aulas sobre estruturas de controle de decisão

Ruben Carlo Benante
Autor1
Thiago De Azevedo Cavendish
Autor3
Ulisses Mosart Sobrinho
Autor5
Autor6

5 de Novembro de 2021

Resumo

Assunto: Ensino de estruturas de controle de decisão, da Linguagem de Programação C.

Vamos comparar os algoritmos de estruturas de controle de decisão

Local: Escola Politécnica de Pernambuco - UPE/POLI

Órgão Financiador: N/A

Caracterização: Projeto requisito da disciplina de Programação 2, sub-projeto do grupo Doyle

1 Introdução

Esse projeto será composto de duas vídeo aulas sobre o tópico de ensino de estruturas de controle de decisão, da Linguagem de Programação C

- Estrutura de decisão If/Else
- Estrutura de controle Switch

1.1 Função básica

Nós utilizamos as estruturas de decisão (If/Else) quando existem instruções dentro do programa que só devem ser executadas se elas satisfizerem determinadas condições.

.
. Por exemplo :

- Só irei para a praia se não chover.
- Só passarei nesta disciplina se eu obtiver uma média igual ou superior a 7,0 e se a presença for superior ou igual a 70 por cento das aulas.
- A sintaxe da estrutura IF na linguagem C é a seguinte :

H

Tabela 1: Tabela para melhor visualização da estrutura do If

.	if(condição)
Estrutura	chave 1
básica	lista de instruções
.	chave 2

1.2 Estrutura de Decisão IF

- Comando IF = se

.

O algoritmo *acima* trabalha da seguinte maneira :

- A condição é verificada a cada passagem pela estrutura IF. Se a condição for satisfeita (V), então a lista de instruções que se encontra entre chaves será executada.

.

Porém, Se a condição NÃO for satisfeita (F), então serão executadas as instruções existentes logo após o fechamento das chaves.

1.3 Else

1.3.1 Sobre

1.4 Switch

1.4.1 Sobre

A palavra switch, associando ao seu significado do inglês, é um comando que funciona como uma chave de seleção/interruptor, sendo capaz de acionar tanto uma como diversas escolhas.

.

O switch case é um comando utilizado na construção de menus de escolhas (“cases”) para o usuário, o qual diante de um leque de opções, poderá decidir algum dos casos e assim obter uma resposta relativa ao caso selecionado.

1.4.2 Aplicações

O comando “switch(variável)” assemelha seu funcionamento a de conjuntos “if-else”, como demonstrado logo abaixo em um programa cuja principal finalidade é receber e atribuir valores para a variável “valor” e em seguida imprimir na tela uma resposta de acordo com o valor digitado:

```

1 #include <stdio.h>
2
3 int main(void)
4 {
5     int valor;
6
7     printf("Digite um valor de 1 a 3\n");
8     scanf("%i", &valor);
9
10    if (valor == 1)
11        printf("Vitoria!\n");
12    else
13        if(valor == 2)
14            printf("Derrota!\n");
15        else
16            if(valor==3)
17                printf("Empate!\n");
18            else
19                printf("Incorreto, digite um valor de 1 a 3\n");
20
21    return 0;
22 }

```

Figura 1: Exemplo de programa com alguns conjuntos de if-else

Assim como demonstrado acima, é possível criar uma sequência “if-else” em cadeia gerando um conjunto de casos que terão a mesma eficiência do comando switch, no entanto, caso o menu necessite diversos casos, o programa provavelmente ficará desorganizado e estará ocupando bastante espaço de maneira desnecessária.

Agora, apresentando o mesmo programa, só que aplicando o conceito de “switch-case”, ficaria da seguinte maneira:

```

#include <stdio.h>

int main(void)
{
    int valor;

    printf("Digite um valor de 1 a 3\n");
    scanf("%i", &valor);

    switch(valor)
    {
        case 1:
            printf("Vitoria!\n");
            break;
        case 2:
            printf("Derrota!\n");
            break;
        case 3:
            printf("Empate!\n");
            break;
        default:
            printf("Incorreto, digite um valor de 1 a 3\n");
    }

    return 0;
}

```

Figura 2: Aplicando o switch ao invés do if-else

Assim como o “if-else”, switch pode receber tanto um inteiro ou caractere como variável, exemplificado no programa a seguir:

```
1 /*-----Calculadora.c (aplicando o comando switch-case)-----*/
2
3 #include <stdio.h>
4
5 int main()
6 {
7     float num1 = 0, num2 = 0;
8     char operacao = ' ';
9
10
11     printf("Defina a operação que deseja efetuar(+, -, *, /)\n: ");
12     scanf("%c", &operacao);
13
14     printf("\nAgora, indique os numeros que vão realizar tal operacao\n: ");
15     scanf("%f %f", &num1, &num2);
16
17     switch(operacao)
18     {
19         case '+':
20             printf("\nResolucao -> %.2f + %.2f = %.2f\n", num1, num2, num1+num2);
21             break;
22
23         case '-':
24             printf("\nResolucao -> %.2f - %.2f = %.2f\n", num1, num2, num1-num2);
25             break;
26
27         case '*':
28             printf("\nResolucao -> %.2f * %.2f = %.2f\n", num1, num2, num1*num2);
29             break;
30
31         case '/':
32             printf("\nResolucao -> %.2f / %.2f = %.2f\n", num1, num2, num1/num2);
33             break;
34
35         default:
36             printf("\nErro! Operador incorreto...\n");
37     }
38     return 0;
39 }
```

Figura 3: Calculadora

O programa aproveita da praticidade do comando “switch(variável)”, simplificando toda a complexidade que necessitaria de vários “if...else” encadeados.

Nessa calculadora, o usuário deverá digitar a operação seguida dos números a fim de realizar o cálculo em questão:

```
Defina a operação que deseja efetuar(+, -, *, /)
: +

Agora, indique os numeros que vão realizar tal operacao
: 15 5

Resolucao -> 15.00 + 5.00 = 20.00
```

Figura 4: Operação de soma

```
Defina a operação que deseja efetuar(+, -, *, /)
: *

Agora, indique os numeros que vão realizar tal operacao
: 25 4

Resolucao -> 25.00 * 4.00 = 100.00
```

Figura 5: Operação de multiplicação

A instrução “break” no código termina a execução do switch, evitando testar os demais comandos possíveis de forma desnecessária.

O comando “default” serve para exibir uma mensagem caso nenhuma das operações anteriores tenham sido devidamente declaradas:

```
Defina a operação que deseja efetuar(+, -, *, /)
: 2

Agora, indique os numeros que vão realizar tal operacao
: 2 4

Erro! Operador incorreto...
```

Figura 6: Operação indefinida

2 Objetivos

2.1 Objetivo Geral

Descrever o objetivo geral a ser alcançado

2.2 Objetivos Específicos

Listar os objetivos específicos

- Proporcionar tal e tal
- Realizar tal e tal

3 Justificativa

Justificar seu projeto ...

4 Metodologia

Descrever como (por quais métodos) os objetivos serão alcançados.

Esse projeto será composto de duas vídeo aulas sobre o tópico de ensino da Linguagem de Programação C

- Aula sobre as estruturas de decisão If e Else
- Aula sobre a estrutura de controle Switch

O algoritmo é descrito abaixo:

4.1 Equipamentos Necessários

Para realizar este projeto é preciso ter um computador, acesso à internet e tempo ...

4.2 Implementação

Para conseguir blablabla

O algoritmo *Ysort* segue abaixo:

Algoritmo 1 Algoritmo Ysort

```
1: function YSORT(estado) ▷ retorna uma ação
2:   Entradas: estado é a configuração atual do jogo
3:    $v \leftarrow \text{maxvalor}(\text{estado})$ 
4:   retorna a ação  $a$  em sucessores(estado) cujo valor é  $v$ 
5: end function
6: function MAXVALOR(estado) ▷ retorna o valor estático
7:   if fim(estado) then
8:     retorna estatico(estado)
9:   end if
10:   $v \leftarrow -\infty$ 
11:  for todas ações  $a$  nos sucessores(estado) do
12:     $v \leftarrow \max(v, \text{minvalor}(a))$ 
13:  end for
14:  retorna  $v$ 
15: end function
16: function MINVALOR(estado) ▷ retorna o valor estático
17:   if fim(estado) then
18:     retorna estatico(estado)
19:   end if
20:   $v \leftarrow \infty$ 
21:  for todas ações  $a$  nos sucessores(estado) do
22:     $v \leftarrow \min(v, \text{maxvalor}(a))$ 
23:  end for
24:  retorna  $v$ 
25: end function
```

Tabela 2: Tabela de custo de pontos para habilidades

pontos	moedas
8	0
9	1
10	2
11	3
12	4
13	5
14	7
15	9

5 Plano de Trabalho

Esta seção estabelece as atividades a serem realizadas.

6 Cronograma

Em conjunto com a seção de Plano de Trabalho, a seção de cronograma coloca as atividades dispostas numa linha do tempo.

Utilize uma tabela para melhor visualização.

7 Impactos alcançados

7.1 Impacto Científico

Não há impacto científico relevante.

7.2 Impacto Tecnológico

Não há impacto tecnológico relevante.

7.3 Impacto Econômico

Não há impacto econômico relevante.

7.4 Impacto Social

O projeto visa contribuir com o aprendizado das futuras gerações da sociedade de forma que... bla ... bla... blal

7.5 Impacto Ambiental

Não há impacto ambiental relevante.

8 Conclusão

- O Grupo Doyle terá como objetivo na execução dessas aulas ensinar o controle de decisão, da linguagem C, aos discentes interessados. Contará com sua exposição de ensino gravada que será disponibilizada e a elaboração de relatórios a fim de cumprir com os aspectos estabelecidos no Plano de Trabalho.

8.1 Concluindo

O que mais o seu projeto agrega? O que é transferido? De onde vem? Para onde vai?

9 Resultados Esperados

Os resultados mostrados na tabela 2 demonstram ...

Concluimos, com base nos estudos e testes coletados sobre os algoritmos de ordenação propostos, que para fins educacionais, o algoritmo *BubbleSort* é mais indicado devido a sua simples implementação, cabendo então para o *QuickSort* ser o mais indicado entre os dois, quando requer uma demanda em menor tempo e com mais eficiência. .

- De acordo com [1] e este é o fim do artigo.

Referências Bibliográficas

Referências

- [1] BENANTE, R. C. *Geração de Trajetórias de Estados por Mapas Auto-organizáveis com Topologia Dinâmica*. Doutorado em ciências da computação, Universidade Federal de Pernambuco, Recife, 2008.