
Raciocínio Algorítmico e Fundamentos da Computação

— Danilo Rios —
21/03/2025

Constante

- O que é?

Constante

- O que é?
 - É o oposto de variável
 - Algo que não muda

Constante

- É algo no código que guarda alguma informação
- Porém essa informação é fixa
 - Não dá para trocar

Constante

- #define <nome da constante> <valor>
 - Nome da constante todo em MAIÚSCULO para diferenciar de variável

```
1  #include <stdio.h>
2
3  #define PI 3.14
4
5  int main() {
6      printf("O valor de PI = %f\n", PI);
7      printf("O valor de PI com 2 casas decimais = %.2f\n", PI);
8      return 0;
9  }
```

Código em texto

```
#include <stdio.h>
```

```
#define PI 3.14
```

```
int main() {
```

```
    printf("O valor de PI = %f\n", PI);
```

```
    printf("O valor de PI com 2 casas decimais = %.2f\n", PI);
```

```
    return 0;
```

```
}
```

Exercício 16

- Criar um código
 - Define uma constante com a quantidade de horas do dia
 - Pede para digitar uma quantidade de dias
 - Calcula quantas horas são a quantidade de dias digitados
 - Imprimir o texto
 - <dias digitados> dias sao equivalente a <quantidade de horas> horas
- Enviar o exercício pelo Moodle
 - Conteúdo: arquivo .c

Variável

- O que é?

Variável

- O que é?
 - Não aprendemos sobre variável?
 - Tem os tipos
 - int, float, double, char
 - Declaramos, atribuímos valor e exibimos o valor armazenado

Contexto da variável

- O que é um contexto?

Contexto da variável

- O que é um contexto?
 - É a delimitação de algo

Contexto da variável

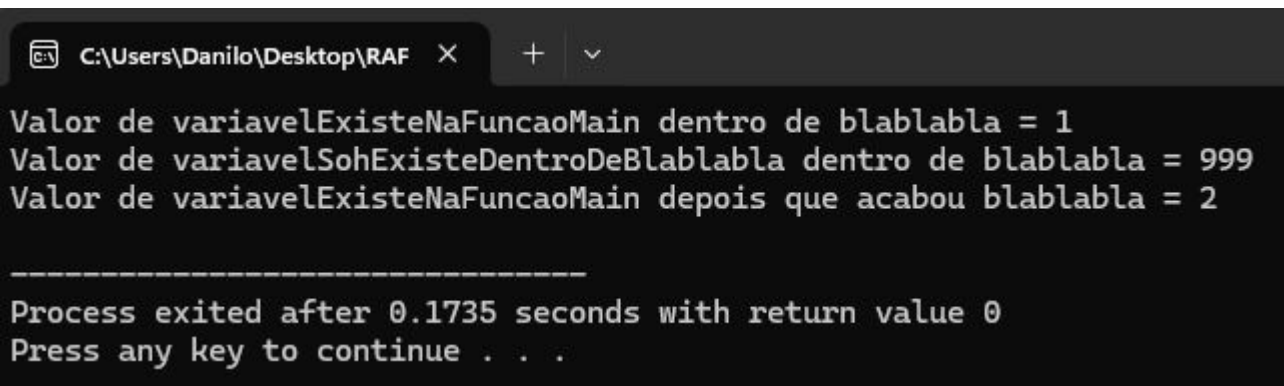
- É onde a variável existe e pode ser utilizada
- Delimitação acontece com as chaves { }

```
1  #include <stdio.h>
2
3  int main() {
4
5      int variavelExisteNaFuncaoMain = 1;
6
7      blablabla{ //nas próximas aulas veremos o que pode ser esse blablabla
8          printf("Valor de variavelExisteNaFuncaoMain dentro de blablabla = %d\n",variavelExisteNaFuncaoMain);
9
10         variavelExisteNaFuncaoMain = 2;
11
12         int variavelSohExisteDentroDeBlablabla = 999;
13
14         printf("Valor de variavelSohExisteDentroDeBlablabla dentro de blablabla = %d\n",variavelSohExisteDentroDeBlablabla);
15     }
16
17     printf("Valor de variavelExisteNaFuncaoMain depois que acabou blablabla = %d\n",variavelExisteNaFuncaoMain);
18
19     //printf("Valor de variavelExisteDentroDeBlablabla depois que acabou blablabla = %d\n",variavelExisteDentroDeBlablabla);
20
21     return 0;
22 }
```

```

1  #include <stdio.h>
2
3  int main() {
4
5      int variavelExisteNaFuncaoMain = 1;
6
7      blablabla{ //nas próximas aulas veremos o que pode ser esse blablabla
8          printf("Valor de variavelExisteNaFuncaoMain dentro de blablabla = %d\n",variavelExisteNaFuncaoMain);
9
10         variavelExisteNaFuncaoMain = 2;
11
12         int variavelSohExisteDentroDeBlablabla = 999;
13
14         printf("Valor de variavelSohExisteDentroDeBlablabla dentro de blablabla = %d\n",variavelSohExisteDentroDeBlablabla);
15     }
16
17     printf("Valor de variavelExisteNaFuncaoMain depois que acabou blablabla = %d\n",variavelExisteNaFuncaoMain);
18
19     //printf("Valor de variavelExisteDentroDeBlablabla depois que acabou blablabla = %d\n",variavelExisteDentroDeBlablabla);
20
21     return 0;
22 }

```



```

C:\Users\Danilo\Desktop\RAF X + v

Valor de variavelExisteNaFuncaoMain dentro de blablabla = 1
Valor de variavelSohExisteDentroDeBlablabla dentro de blablabla = 999
Valor de variavelExisteNaFuncaoMain depois que acabou blablabla = 2

-----
Process exited after 0.1735 seconds with return value 0
Press any key to continue . . .

```

```
1  #include <stdio.h>
2
3  int main() {
4
5      int variavelExisteNaFuncaoMain = 1;
6
7      blablabla{ //nas próximas aulas veremos o que pode ser esse blablabla
8          printf("Valor de variavelExisteNaFuncaoMain dentro de blablabla = %d\n",variavelExisteNaFuncaoMain);
9
10         variavelExisteNaFuncaoMain = 2;
11
12         int variavelSohExisteDentroDeBlablabla = 999;
13
14         printf("Valor de variavelSohExisteDentroDeBlablabla dentro de blablabla = %d\n",variavelSohExisteDentroDeBlablabla);
15     }
16
17     printf("Valor de variavelExisteNaFuncaoMain depois que acabou blablabla = %d\n",variavelExisteNaFuncaoMain);
18
19     printf("Valor de variavelExisteDentroDeBlablabla depois que acabou blablabla = %d\n",variavelExisteDentroDeBlablabla);
20
21     return 0;
22 }
```

- O que vai acontecer?

Contexto da variável

- Erro na compilação do código!

```
17 printf("Valor de variavelExisteNaFuncaoMain depois que acabou blablabla = %d\n",variavelExisteNaFuncaoMain);
18
19 printf("Valor de variavelExisteDentroDeBlablabla depois que acabou blablabla = %d\n",variavelExisteDentroDeBlablabla);
20
21 public int __cdecl printf (const char * __restrict__ _Format, ...)
22 }
```

Compiler (3) Resources Compile Log Debug Find Results Close			
Line	Col	File	Message
		C:\Users\Danilo\Desktop\RAFC ...	In function 'main':
19	87	C:\Users\Danilo\Desktop\RAFC 2...	[Error] 'variavelExisteDentroDeBlablabla' undeclared (first use in this function)
19	87	C:\Users\Danilo\Desktop\RAFC 2...	[Note] each undeclared identifier is reported only once for each function it appears in

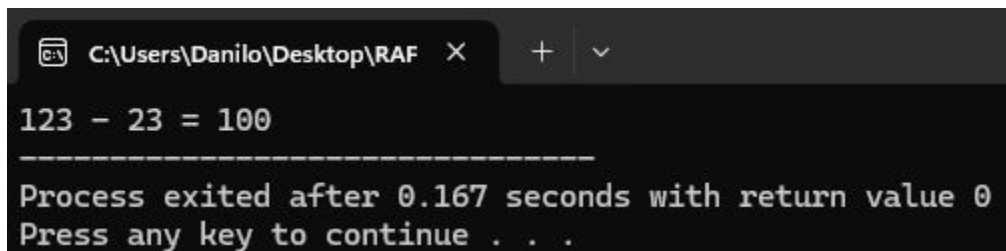
Contexto da variável

```
1  #include <stdio.h>
2
3  int variavelGlobal = 123;
4
5  int main() {
6      int variavelNaFuncaoMain = 23;
7      int resultadoSubtracao = variavelGlobal - variavelNaFuncaoMain;
8
9      printf("%d - %d = %d", variavelGlobal, variavelNaFuncaoMain, resultadoSubtracao);
10
11     return 0;
12 }
13
```

- Funciona ou vai dar erro?

Contexto da variável

```
1  #include <stdio.h>
2
3  int variavelGlobal = 123;
4
5  int main() {
6
7      int variavelNaFuncaoMain = 23;
8      int resultadoSubtracao = variavelGlobal - variavelNaFuncaoMain;
9
10     printf("%d - %d = %d", variavelGlobal, variavelNaFuncaoMain, resultadoSubtracao);
11
12     return 0;
13 }
```



C:\Users\Danilo\Desktop\RAF x + v

123 - 23 = 100

Process exited after 0.167 seconds with return value 0
Press any key to continue . . .

Contexto da variável

- Então é só declarar tudo global?

Contexto da variável

- Então é só declarar tudo global?
 - O professor que me ensinou a linguagem C tinha a seguinte frase
 - “Quem utiliza variável global vai para o inferno da computação”
 - E esse é 1 de 3 motivos que te levam pra lá

Contexto da variável

- Então é só declarar tudo global?
 - Se todas as variáveis forem globais
 - Além de arriscar ir para o inferno da computação ...
 - Não vai ter o “problema” de utilizar uma variável e ela não existir
 - E acabaram as vantagens

Contexto da variável

- Quais são os problemas/desvantagens de ter todas variáveis globais?
 - Alocação de memória
 - Todas as variáveis globais vão estar alocando espaço na memória durante toda a execução do programa
 - Quantidade de variáveis
 - Tem que declarar muitas variáveis
 - Se tentar aproveitar alguma variável pode ser que ela era utilizada depois, mas você esqueceu, mudou o valor dela e o programa tem a saída errada
 - Alteração no valor
 - Qualquer parte do código é capaz de alterar o valor da variável indevidamente

Contexto da variável

- Não é difícil criar a variável no contexto certo
- Podemos ter variável global no código?
 - Sim, mas só se for extremamente necessário

Convenção

- O que é?
 - Sem ser um evento

Convenção

- O que é?
 - É um padrão adotado pelas pessoas

Convenção

- Em TI nós temos algumas convenções
 - Alguns padrões que seguimos

Convenção

- Todos os códigos que já apareceram estão nesta convenção que vamos conversar
- Como devemos escrever o nosso código?
 - Pelo menos as declarações que fazemos

Convenção

- Utilizamos em C e em outras linguagens
- camelCase
 - Caso do camelo



Convenção - camelCase

- As variáveis e as funções do código
 - Começam sempre com letra minúscula
 - Quando o nome possui mais de 1 palavra, a 1ª letra da 2ª, 3ª... palavra é maiúscula
- Ex.:
 - main
 - ano
 - anoAtual
 - declaracaoCom varias Palavras

Indentação

- O que é?

Indentação

- O que é?
 - Dar um afastamento

Indentação

- Deixa o código mais legível, mais fácil de entender
- Abriu a chave { as próximas linhas devem ter 1 tab a mais
- Fechou a chave } volta 1 tab

Indentação

```
1  #include <stdio.h>
2
3  int main() {
4
5      int variavelExisteNaFuncaoMain = 1;
6
7      blablabla{ //nas próximas aulas veremos o que pode ser esse blablabla
8          printf("Valor de variavelExisteNaFuncaoMain dentro de blablabla = %d\n",variavelExisteNaFuncaoMain);
9
10         variavelExisteNaFuncaoMain = 2;
11
12         int variavelSohExisteDentroDeBlablabla = 999;
13
14         printf("Valor de variavelSohExisteDentroDeBlablabla dentro de blablabla = %d\n",variavelSohExisteDentroDeBlablabla);
15     }
16
17     printf("Valor de variavelExisteNaFuncaoMain depois que acabou blablabla = %d\n",variavelExisteNaFuncaoMain);
18
19     printf("Valor de variavelExisteDentroDeBlablabla depois que acabou blablabla = %d\n",variavelExisteDentroDeBlablabla);
20
21     return 0;
22 }
```

Exercício 17

- Criar um código, desse exercício em diante, com indentação e com a padronização camelCase
 - Definir uma constante com o preço da bola de futebol com o valor 49,90
 - Imprimir o texto
 - O valor da bola de futebol é <valor>. Quantas voce quer comprar?
 - Pegar a quantidade de bolas
 - Imprimir o texto
 - O valor total da sua compra eh de <valor da compra>
 - Se pagar no dinheiro tem 10% de desconto e a compra fica com o valor de <valor com desconto>
- Obs.: Para escrever o caractere % tem que colocar %% no printf
- Enviar o exercício pelo Moodle
 - Conteúdo: arquivo .c

Perguntas?

Obrigado!

Até a próxima aula!