
Raciocínio Algorítmico e Fundamentos da Computação

— Danilo Rios —
14/03/2025

Mais 1 passo...

- Precisamos obter/receber informações de quem está utilizando o nosso programa

Entrada de dados através do teclado

- Função scanf

```
1  #include <stdio.h>
2
3  int main() {
4
5      int numeroInt;
6      float numeroFloat;
7      double numeroDouble;
8      char caractere;
9
10     printf("Digite um inteiro: ");
11     scanf("%d", &numeroInt);
12     printf("\n\nDigite um float: ");
13     scanf("%f", &numeroFloat);
14     printf("\n\nDigite um double: ");
15     scanf("%lf", &numeroDouble);
16     printf("\n\nDigite um caractere: ");
17     scanf("%c", &caractere);
18
19     printf("\n\nDigitou: %d, %f, %f, %c", numeroInt, numeroFloat, numeroDouble, caractere);
20
21     return 0;
22 }
```

```
1  #include <stdio.h>
2
3  int main() {
4
5      int numeroInt;
6      float numeroFloat;
7      double numeroDouble;
8      char caractere;
9
10     printf("Digite um inteiro: ");
11     scanf("%d", &numeroInt);
12     printf("\n\nDigite um float: ");
13     scanf("%f", &numeroFloat);
14     printf("\n\nDigite um double: ");
15     scanf("%lf", &numeroDouble);
16     printf("\n\nDigite um caractere: ");
17     scanf("%c", &caractere);
18
19     printf("\n\nDigitou: %d, %f, %f, %c", numeroInt, numeroFloat, numeroDouble, caractere);
20
21     return 0;
22 }
```

- scanf(tipo informação, & variável);
 - Tipo informação:
 - %d - int
 - %f - float
 - %lf - double
 - %c - char
 - & - e comercial
 - Variável que vai armazenar a informação

& variável

- O que significa?

& variável

- O que significa?
 - A informação lida do teclado vai ser armazenada no endereço de memória da variável

scanf

- O scanf não atribui um valor para a variável
 - `int x = 10;`
 - `int x = scanf("%d");`
- Ele coloca a informação direto na memória do computador
 - Por isso que precisa do endereço de memória
- Quando o programa vai utilizar a variável, a variável “pega” a informação que está guardada no endereço de memória que é dela e está com a nova informação

Endereço de memória

```
int int1;  
int int2;  
float float1;  
float float2;  
double double1;  
double double2;  
char char1;  
char char2;
```

```
int1 = 000000000062FE4C  
int2 = 000000000062FE48  
float1 = 000000000062FE44  
float2 = 000000000062FE40  
double1 = 000000000062FE38  
double2 = 000000000062FE30  
char1 = 000000000062FE2F  
char2 = 000000000062FE2E
```

Endereço de memória

```
int int1;  
int int2;  
float float1;  
float float2;  
double double1;  
double double2;  
char char1;  
char char2;
```

```
int1 = 000000000062FE4C } 4 bytes  
int2 = 000000000062FE48 } 4 bytes  
float1 = 000000000062FE44 } 4 bytes  
float2 = 000000000062FE40 } 4 bytes  
double1 = 000000000062FE38 } 8 bytes  
double2 = 000000000062FE30 } 8 bytes  
char1 = 000000000062FE2F } 1 byte  
char2 = 000000000062FE2E } 1 byte
```

Endereço de memória

- Sempre estão corretos?

Endereço de memória

- Sempre estão corretos?
 - Nem sempre...

```
int int1;  
int int2;  
float float1;  
double double1;  
char char1;  
char char2;
```

```
int1 = 0000000000062FE4C } 4 bytes  
int2 = 0000000000062FE48 } 4 bytes  
float1 = 0000000000062FE44 } 12 bytes  
double1 = 0000000000062FE38 } 1 byte  
char1 = 0000000000062FE37 } 1 byte  
char2 = 0000000000062FE36 }
```

Endereço de memória

```
int int1;  
int int2;  
float float1;  
double double1;  
char char1;  
char char2;
```

```
int1 = 000000000062FE4C  
int2 = 000000000062FE48  
float1 = 000000000062FE44  
double1 = 000000000062FE38  
char1 = 000000000062FE37  
char2 = 000000000062FE36
```

} 12 bytes ?

No slide sobre tipo
de variável não
estava que double
são 8 bytes !?!?

Endereço de memória

- Sempre a variável vai ter o tamanho correto
- O que pode acontecer é ter um endereço de memória que são pulados

Endereço de memória

- Compilador ao transformar o código em linguagem de máquina pode colocar esses “espaços em branco” para otimizar/melhorar o acesso à informação

Endereço de memória

- 1 byte = 8 bits
- SO 32 bits = 4 bytes
 - Cada “grupo” de memória são 4 bytes
- SO 64 bits = 8 bytes
 - Cada “grupo” de memória são 8 bytes
- Então no exemplo ao invés de dividir o double de 8 bytes entre 2 “grupos” de memória o compilador dá o “espaço” e coloca a informação num único grupo

Endereço de memória

Variável	Hexadecimal	Decimal	Dividido por 8
	62FE50	6487632	810954
int1	62FE4C	6487628	810953.5
int2	62FE48	6487624	810953
float1	62FE44	6487620	810952.5
double1	62FE38	6487608	810951
char1	62FE37	6487607	810950.875
char2	62FE36	6487606	810950.75

Endereço de memória

```
int1 = 000000000062FE4C
int2 = 000000000062FE48
float1 = 000000000062FE44
double1 = 000000000062FE38
char1 = 000000000062FE37
char2 = 000000000062FE36
```

62FE50	62FE51	62FE52	62FE53	62FE54	62FE55	62FE56	62FE57
62FE48	62FE49	62FE4A	62FE4B	62FE4C	62FE4D	62FE4E	62FE4F
int2	int2	int2	int2	int1	int1	int1	int1
62FE40	62FE41	62FE42	62FE43	62FE44	62FE45	62FE46	62FE47
				float	float1	float1	float1
62FE38	62FE39	62FE3A	62FE3B	62FE3C	62FE3D	62FE3E	62FE3F
double1	double1	double1	double1	double1	double1	double1	double1
62FE30	62FE31	62FE32	62FE33	62FE34	62FE35	62FE36	62FE37
						char2	char1

Endereço de memória

- Obs.: Quem quiser fazer em casa
- Utilizando %p vai imprimir a informação em hexadecimal
- <variavel> é o endereço de memória da variável

```
printf("int1 = %p\n",&int1);
```

Continuando...

- Depois desse enorme parênteses para falar de memória
- Vamos executar o código que tem o scanf

```
1  #include <stdio.h>
2
3  int main() {
4
5      int numeroInt;
6      float numeroFloat;
7      double numeroDouble;
8      char caractere;
9
10     printf("Digite um inteiro: ");
11     scanf("%d", &numeroInt);
12     printf("\n\nDigite um float: ");
13     scanf("%f", &numeroFloat);
14     printf("\n\nDigite um double: ");
15     scanf("%lf", &numeroDouble);
16     printf("\n\nDigite um caractere: ");
17     scanf("%c", &caractere);
18
19     printf("\n\nDigitou: %d, %f, %f, %c", numeroInt, numeroFloat, numeroDouble, caractere);
20
21     return 0;
22 }
```

Código em texto

```
#include <stdio.h>

int main() {

    int numeroInt;
    float numeroFloat;
    double numeroDouble;
    char caractere;

    printf("Digite um inteiro: ");
    scanf("%d", &numeroInt);
    printf("\n\nDigite um float: ");
    scanf("%f", &numeroFloat);
    printf("\n\nDigite um double: ");
    scanf("%lf", &numeroDouble);
    printf("\n\nDigite um caractere: ");
    scanf("%c", &caractere);

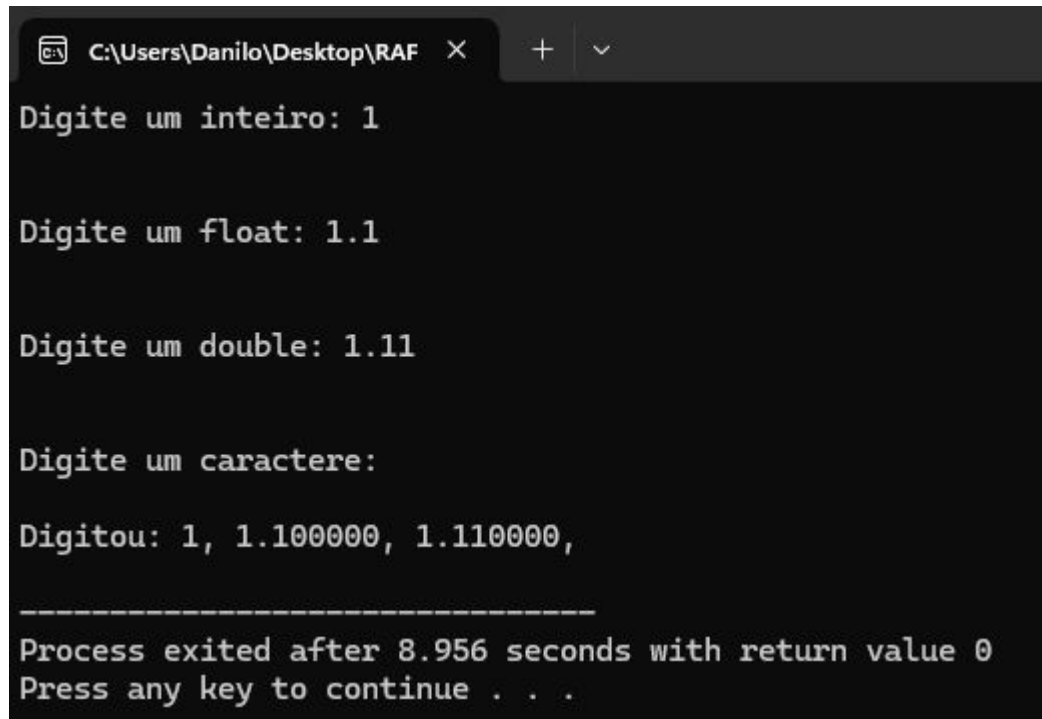
    printf("\n\nDigitou: %d, %f, %f, %c", numeroInt, numeroFloat, numeroDouble, caractere);

    return 0;

}
```

Entrada de dados através do teclado

- Não esperou digitar o caractere !?!?
 - O que aconteceu?



```
C:\Users\Danilo\Desktop\RAF x + v
Digite um inteiro: 1
Digite um float: 1.1
Digite um double: 1.11
Digite um caractere:
Digitou: 1, 1.100000, 1.110000,
-----
Process exited after 8.956 seconds with return value 0
Press any key to continue . . .
```

Entrada de dados através do teclado

- O que aconteceu?
 - Existe o buffer de entrada que possui “lixo” armazenado
 - Esse “lixo” é o que foi lido pelo scanf que ia pegar o caractere digitado

Entrada de dados através do teclado

- Como corrigir?
 - Precisamos limpar esse buffer de entrada para não ter o “lixo”
 - Que é a leitura do `\n` do enter, no caso em questão

Entrada de dados através do teclado

- Como corrigir?
 - Precisamos limpar esse buffer de entrada para não ter o “lixo”
 - Que é a leitura do `\n` do enter, no caso em questão
- Função `getchar()`
 - Lê 1 caractere do buffer de entrada

```
1  #include <stdio.h>
2
3  int main() {
4
5      int numeroInt;
6      float numeroFloat;
7      double numeroDouble;
8      char caractere;
9
10     printf("Digite um inteiro: ");
11     scanf("%d", &numeroInt);
12     printf("\n\nDigite um float: ");
13     scanf("%f", &numeroFloat);
14     printf("\n\nDigite um double: ");
15     scanf("%lf", &numeroDouble);
16
17     getchar();
18
19     printf("\n\nDigite um caractere: ");
20     scanf("%c", &caractere);
21
22     printf("\n\nDigitou: %d, %f, %f, %c", numeroInt, numeroFloat, numeroDouble, caractere);
23
24     return 0;
25 }
```

Código em texto

```
#include <stdio.h>

int main() {

    int numeroInt;
    float numeroFloat;
    double numeroDouble;
    char caractere;

    printf("Digite um inteiro: ");
    scanf("%d", &numeroInt);
    printf("\n\nDigite um float: ");
    scanf("%f", &numeroFloat);
    printf("\n\nDigite um double: ");
    scanf("%lf", &numeroDouble);

    getchar();

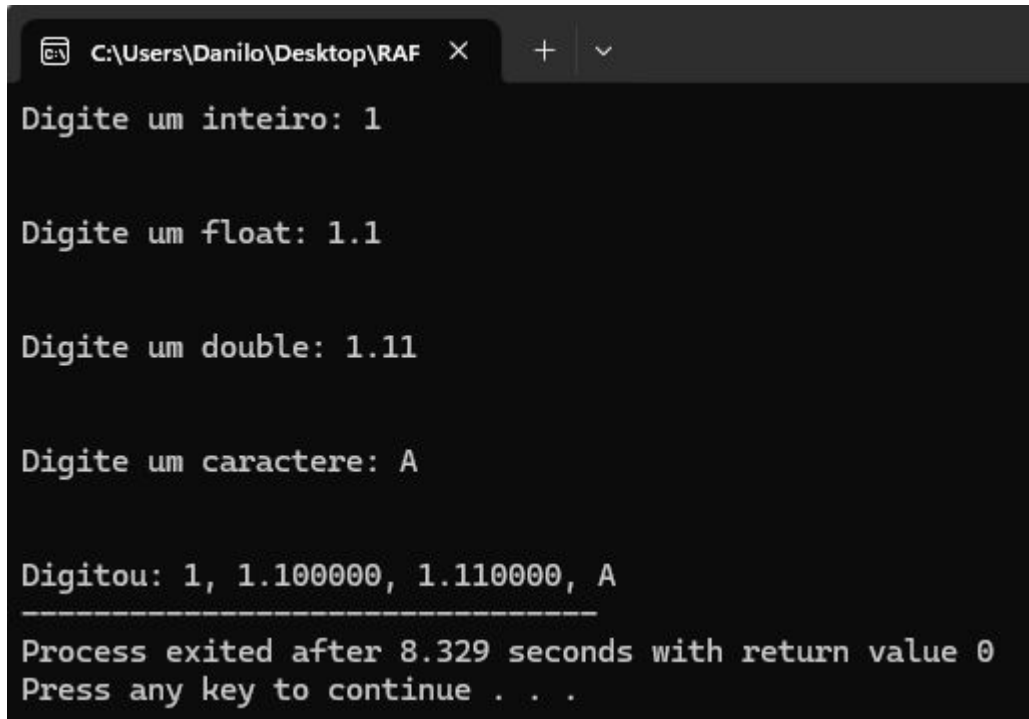
    printf("\n\nDigite um caractere: ");
    scanf("%c", &caractere);

    printf("\n\nDigitou: %d, %f, %f, %c", numeroInt, numeroFloat, numeroDouble, caractere);

    return 0;
}
```

Entrada de dados através do teclado

- Deu certo!



```
C:\Users\Danilo\Desktop\RAF >
Digite um inteiro: 1

Digite um float: 1.1

Digite um double: 1.11

Digite um caractere: A

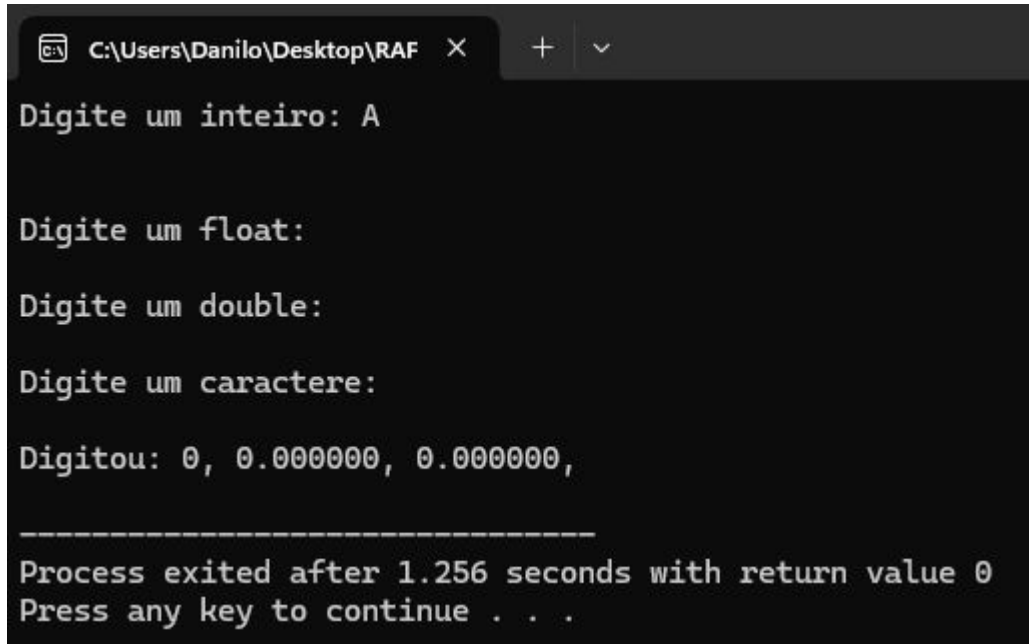
Digitou: 1, 1.100000, 1.110000, A
-----
Process exited after 8.329 seconds with return value 0
Press any key to continue . . .
```

Vamos fazer outro combinado

- Já combinamos que não vamos utilizar acentuação
- Nosso outro combinado é: Só vamos escrever o que for pedido
 - Pediu para digitar um número
 - Vamos digitar um número inteiro
 - Pediu para digitar um número com casa decimal
 - Vamos digitar um número inteiro ou com casa decimal usando o ponto
 - Pediu para digitar um caractere
 - Vamos digitar 1 caractere

Vamos fazer outro combinado

- Não seguiu o combinado => Estragou a brincadeira



```
C:\Users\Danilo\Desktop\RAF X + v
Digite um inteiro: A

Digite um float:

Digite um double:

Digite um caractere:

Digitou: 0, 0.000000, 0.000000,

-----
Process exited after 1.256 seconds with return value 0
Press any key to continue . . .
```

Comentários

- São informações no código fonte
 - São ignoradas pelo compilador
 - Servem para deixar uma explicação e auxiliar a entender o código

```
3  [ ] int main() {  
4  
5      //comentario de 1 linha  
6  
7  
8      /*  
9          Comentario  
10         com  
11         varias  
12         linhas  
13     */  
14  
15     return 0;  
16 }
```


Exercício 14

- Criar um código
 - Pedir para digitar o ano de nascimento
 - Imprimir o texto
 - No final de <ano atual> voce vai ter <idade> anos.
- Enviar o exercício pelo Moodle
 - Conteúdo: arquivo .c

Exercício 15

- Criar um código
 - Pedir o dia de aniversário
 - Pedir o número do mês de aniversário
 - Pedir a letra inicial do nome
 - Pedir a letra inicial do sobrenome
 - Imprimir o texto
 - Que legal! <letra inicial do nome> <letra inicial do sobrenome> voce faz aniversario em <dia>/<mes>.
 - De qual ano?
 - Pegar o ano
 - Imprimir o texto
 - Entao vc nasceu em <ano>. Eu sou mais novo que voce e nasci 3 anos depois em <ano calculado>
- Enviar o exercício pelo Moodle
 - Conteúdo: arquivo .c

Perguntas?

Obrigado!

Até a próxima aula!