Programação em C, variáveis, tipos, operadores, E/S

Prof. Fernando Figueira (adaptado do material do Prof. Rafael Beserra Gomes)

UFRN

Material compilado em 27 de agosto de 2025. Licença desta apresentação:



http://creativecommons.org/licenses/

- Desenvolvimento inicial por Dennis Ritchie por volta de 1970
- Qual a sintaxe da linguagem C?
- Em 1983, ANSI (American National Standards Institute) padronizou C (ANSI C)
- Padrões seguintes pelo ISO: C90, C99, C11
- O gnu/gcc (5.1.0 ou superior) usa por padrão o padrão gnu11 (C11 com extensões)
- Cada compilador pode incluir extensões (pode estar fora do padrão e não funcionar em outro compilador!)

Estrutura inicial de um programa escrito em C:

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5
6    printf("Mensagem escrita na tela\n");
7    printf("Uma outra mensagem\n");
8
9    return 0;
10 }
```

- ► Cabeçalho
- Definição de uma função principal chamada main: instruções e retorno

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5
6    printf("Mensagem escrita na tela\n");
7    printf("Uma outra mensagem\n");
8
9    return 0;
10 }
```

- Instruções encerradas com ;
- Quebras de linha e indentação (espaçamento em relação à margem) facilitam a legibilidade do código-fonte
- Comentários podem ser feitos com:
 - // para uma única linha
 - /* ... */ para múltiplas linhas

Forma mais simples (gera executável a.out) gcc nomeCodigo.c

gcc nomeCodigo.c -o nomeExecutavel -std=padrao -O3
Há centenas de opções para o compilador gcc, algumas:

- -o: especifica o nome do executável (se omitir: a.out)
- -std: padrão a ser utilizado na compilação (se omitir: gnu11)
- -pedantic: alerta se há algo fora do padrão especificado
- -O3: habilita algumas otimizações de código

Para esse caso, a ordem das caixas coloridas não importa.

IDE: *integrated development environment* auxiliam a programação de computadores

- Texto colorido para melhorar a legibilidade do código
- Atalhos para compilação, acesso fácil para configurações
- Recursos para refatoração, debug, etc

Exemplo de editores/IDE:

- gedit (mais básico)
- Geany
- Code::Blocks
- Sublime
- Vi/Vim (mais avançado)

Variáveis e tipos

Exemplo de dados na memória:

0xbffff22c	0	0	0	0	0	1	0	1	5 inteiro curto
0xbffff22d	0	1	0	0	0	0	1	0	B caractere
0xbffff22e	0	1	0	0	0	0	1	1	C caractere
0xbffff22f	1	1	0	1	1	1	0	1	3.2 real
0xbffff230	1	1	0	0	1	1	0	0	
0xbffff231	0	1	0	0	1	1	0	0	
0xbffff232	0	1	0	0	0	0	0	0	
0xbffff233	0	1	0	0	0	0	0	1	A caractere
0xbffff234	0	1	0	0	0	0	1	0	B caractere
0xbffff235	0	1	0	0	0	0	1	1	C caractere
0xbffff236	0	1	0	0	0	0	1	1	A caractere
0xbffff237	1	1	1	1	1	0	1	1	-5 inteiro curto
0xbffff238	0	0	0	0	0	0	0	0	1 inteiro
0xbffff239	0	0	0	0	0	0	0	0	
0xbffff23a	0	0	0	0	0	0	0	0	
0xbffff23b	0	0	0	0	0	0	0	1	
	0xbffff22d 0xbffff22e 0xbffff22e 0xbffff230 0xbffff231 0xbffff233 0xbffff233 0xbffff234 0xbffff235 0xbffff236 0xbffff236 0xbffff237 0xbffff238 0xbffff239 0xbffff23a	0xbfff22d 0 0xbfff22e 0 0xbfff22e 1 2x 0xbfff230 1 0xbfff231 0 0xbfff231 0 0xbfff233 0 0xbfff233 0 0xbfff234 0 0xbfff235 0 0xbfff236 0 0xbfff236 0 0xbfff237 1 0xbfff238 0 0xbfff239 0 0xbfff23a 0	Oxbfff22d 0 1 Oxbfff22e 0 1 Oxbfff22e 0 1 Oxbfff22e 1 1 Oxbfff230 1 1 Oxbfff231 0 1 Oxbfff232 0 1 Oxbfff233 0 1 Oxbfff234 0 1 Oxbfff235 0 1 Oxbfff236 0 1 Oxbfff237 1 1 Oxbfff238 0 0 Oxbfff238 0 0 Oxbfff238 0 0	Oxbfff22d 0 1 0 Oxbfff22e 0 1 1 0 Oxbfff22e 1 1 1 0 Oxbfff22f 1 1 1 0 Oxbfff231 0 1 0 Oxbfff231 0 1 0 Oxbfff232 0 1 0 Oxbfff233 0 1 0 Oxbfff234 0 1 0 Oxbfff235 0 1 0 Oxbfff236 0 1 0 Oxbfff236 0 1 0 Oxbfff237 1 1 1 1 Oxbfff238 0 0 0 Oxbfff238 0 0 0	Oxbifff22d	Oxbifff22d	Oxbifff22d	Oxbifff22d	Oxbifff22d 0 1 0 0 0 0 0 1 1 0 Oxbifff22e 0 1 0 0 0 0 0 1 1 1 Oxbifff22e 1 1 0 0 0 0 0 1 1 1 0 Oxbifff23e 0 1 0 0 0 0 0 0 0 1 1 0 Oxbifff23e 0 1 0 0 0 0 0 0 1 0 Oxbifff23e 0 1 0 0 0 0 0 0 1 0 Oxbifff23e 0 1 0 0 0 0 0 0 1 0 Oxbifff23e 0 1 0 0 0 0 0 0 1 1 Oxbifff23e 0 1 0 0 0 0 0 0 1 1 Oxbifff23e 0 1 0 0 0 0 0 1 1 0 Oxbifff23e 0 1 0 0 0 0 0 1 1 0 Oxbifff23e 0 1 0 0 0 0 0 1 1 1 Oxbifff23e 0 1 0 0 0 0 0 0 1 1 1 Oxbifff23e 0 1 0 0 0 0 0 0 1 1 1 Oxbifff23e 0 1 0 0 0 0 0 0 1 1 1 Oxbifff23e 0 1 0 0 0 0 0 0 0 1 1 1 Oxbifff23e 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Oxbiff23e 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Endereço na memória (em hexadecimal)

- A variável representa um dado variável (pode ser alterado) na memória
- Cada variável possui:
 - tipo (ex.: int, float, string, boolean)
 - endereço: posição do dado na memória, na qual há uma sequência de bits1 representando um valor
 - identificador, evitando que tenhamos que saber o endereço dos dados

¹a quantidade de bits depende do tipo da variável



Em relação a C:

é necessário declarar variáveis

```
#include <stdio.h>
2
3
  int main() {
4
5
       int x;
6
       float y;
       int z = 323:
8
       int a, b = 2, c;
9
10
       return 0:
11
```

- somente pode usar uma variável depois de declará-la
- pode atribuir um valor inicial para cada variável (inicialização)²
- pode declarar mais de uma variável na mesma linha, desde que sejam do mesmo tipo

²Uma boa regra é procurar sempre inicializar as variáveis. Apesar de que em testes você poderá constatar um valor zero inicial para as variáveis, nem sempre será o caso. 🦠 a 🖎

Experimente o seguinte código:

```
#include <stdio.h>
2
  int main() {
4
5
       int a = 3;
6
       printf("Valor da variavel a: %d\n", a);
7
       printf("Endereco da variavel a: %p\n", &a);
8
       a = a + 2;
9
       printf("Valor da variavel a: %d\n", a);
10
11
       return 0;
12
```

Em relação a C:

tipos primitivos principais: char, int, long long, float, double

Tipo	Exemplo de valor			
char ³	'a', 'z', 'A', 'T' (tabela ascii)			
int	1232, 502, -39328			
long long int	2395828482, -2392542832			
float	3.242, 52002.1			
double	3.242, 52002.1			

 as versões unsigned utilizam a variável somente para valores não negativos

³um caractere tem valor inteiro de acordo com a tabela ascii

Strings são sequências de caracteres

- uma string é representa entre aspas duplas
 - 1 "eis uma string"
- é possível armazenar uma string em C, mas veremos mais adiante na disciplina

Identificadores em C:

- NÃO comece com dígitos
- NÃO use espaços, acentos, ç, ou caracteres diferentes de a..z A..Z
- ▶ NÃO use palavras-chave da linguagem (ex.: int, return, float, if, for)
- são case-sensitive, exemplo: variavel é diferente de VaRiAVel

Recomendações:

- iniciar com letra minúscula para variáveis
- utilize um identificador que represente bem o significado da variável
- se for composto por vários nomes, comece cada nome (exceto o primeiro) com letra maiúscula (ex.: anguloTeste)
- NÃO use identificadores longos demais (ex.: valorInteiroASerDigitadoPeloUsuario)

Entrada e saída de dados

Através da função printf (stdio.h):

```
1 printf("Uma mensagem\n");
2 printf("Estou programando");
3 printf("em C\n");
```

- ► É comum dizer que o programa escreve na tela
- Você pode inserir valores na string a ser escrita:

```
1 printf("Mes = %d e Ano = %d\n", 2, 2017);
```

haverá tantos argumentos quantos especificadores de formato na string (%), atribuídos na mesma ordem

```
1 printf("Dia = %d, Mes = %d e Ano = %d\n", 21, 2, 2017);
```

o tipo do dado determina que especificador de formato utilizar

```
1 printf("%d de %s, juros = %f\n", 21, "Fevereiro", 0.13);
```

Clique aqui para conhecer os especificadores de formato da linguagem C.

para escrever na tela números com tantas casas de precisão:

```
1 printf("%.02f", 3.14159265359);
```

Exercício em sala

Escreva um programa em C que declara as seguintes variáveis:

- um caractere com valor 'g' (identificador simboloGravidade)
- ▶ um número real com valor 9.8196 (identificador gravidade)

Em seguida, o programa deve, utilizando o caractere e o número real declarados, escrever na tela a seguinte mensagem:

O valor da gravidade g é: 9.8196 m/s2

► A entrada é realizada através da função scanf (stdio.h):

```
1 int idade;
2 printf("Digite a sua idade: ");
3 scanf("%d", &idade);
```

► É comum dizer que o programa lê do usuário

- haverá tantos argumentos quantos especificadores de formato na string (%), atribuídos na mesma ordem
- o argumento é um endereço de memória (usamos o operador & para obter o endereço de uma variável)

```
1 int idade;
2 printf("Digite a sua idade: ");
3 scanf("%d", &idade);
```

o tipo do dado determina que especificador de formato utilizar

```
1 int idade;
2 float altura;
3 printf("Digite a sua idade: ");
4 scanf("%d", &idade);
5 printf("Digite a sua altura: ");
6 scanf("%f", &altura);
```

você pode utilizar o mesmo scanf para ler mais de um dado

```
1 int idade;
2 float altura;
3 printf("Digite a sua idade e altura: ");
4 scanf("%d %f", &idade, &altura);
```

a mensagem que precede o scanf não é obrigatório, serve apenas para orientar o usuário

```
1 int idade;
2 float altura;
3 scanf("%d %f", &idade, &altura);
```

 o scanf detecta o final de uma entrada para o começo da seguinte a partir de qualquer combinação de espaços, quebras de linha ('\n') ou tabs ('\t') Nas listas de exercícios e nas provas, a não ser que expresso o contrário, você pode assumir que o usuário digita as informações conforme esperado. Por exemplo, para um programa que determina se um número inteiro é primo ou não, você pode assumir que ele digitará um número inteiro e maior que 0.

Exercício em sala

Escreva um programa em C que declara as seguintes variáveis:

- um caractere com valor 'g' (identificador simboloGravidade)
- um número real com valor 9.8196 (identificador gravidade)
- um número inteiro com o tempo (identificador tempo)

O programa deve escrever na tela a mensagem "Digite o tempo de queda: " e ler do usuário o tempo. Em seguida, deve escrever na tela a seguinte mensagem:

O valor da gravidade g é: 9.8196 m/s2

O espaço percorrido pelo objeto em queda livre foi: ... metros Utilize o seguinte para calcular o espaço percorrido:

$$\frac{gt^2}{2}$$