Introdução à Programação C

Introdução à Ciência da Computação I
Prof. Denis F. Wolf

Linguagem C

- Origem de C está associada ao sistema Unix
- Histórico:
  - 1970: Ken Thompson desenvolve B, baseada em BCPL, para o primeiro Unix no DEC PDP-7
  - 1972: Dennis Ritchie projeta a linguagem C, baseada na linguagem B.
  - 1988: o *American National Standard Institute* (ANSI) define o padrão ANSI C
- É considerada uma linguagem de nível médio
- É "case sensitive", ou seja, interpreta como diferentes letras maiúsculas de minúsculas

Programação em C

VARIÁVEIS E TIPOS DE DADOS

Variáveis

- Como armazenar os dados de entrada, fornecidos pelo usuário?
- O que fazer com os resultados das operações?
- Variáveis são elementos que estão associados a posições de memória, cujo objetivo é o armazenamento informações.
- ...por tempo suficiente ao seu processamento

Identificadores

- Nome que fazem referência a elementos tais como as variáveis
- Regras para a definição de identificadores:
  - Na formação do identificador só podem ser utilizados: dígitos, letras (tanto maiúsculas quanto minúsculas) e o caractere de sublinha (_)
  - O identificador deve começar sempre com uma letra ou caractere de sublinha
- Apenas os 31 primeiros caracteres são considerados

Identificadores

- Em C, há diferença entre maiúsculo e minúsculo
  - Exemplo:
  - Nome ≠ nome ≠ NOME
- Não pode ser empregar qualquer uma das palavras reservadas à linguagem C como identificadores

Palavras-chave de C (ANSI)

auto break case char const continue default do double else enum extern float for goto if int long register return short signed sizeof static struct switch typeof union unsigned void volatile while

Variáveis

- Exemplos de nomes de variáveis:

Corretos	Incorretos
Contador	1contador
Teste23	oi!gente
Alto_Paraiso	Alto..Paraíso
__sizeint	_size-int

Tipos de Dados

- O *tipo* de uma variável define os valores que ela pode assumir e as operações que podem ser realizadas com ela
- Descreve a natureza da informação
- Ex:
  - variáveis tipo *int* recebem apenas valores inteiros
  - variáveis tipo *float* armazenam apenas valores reais

Tipos de dados básicos em C

- char: um byte que armazena o código de um caractere do conjunto de caracteres local
- int: um inteiro cujo tamanho depende do processador, tipicamente 16 ou 32 bits
- float: um número real com precisão simples
- double: um número real com precisão dupla

Modificadores de Tipos

- Os modificadores alteram algumas características dos tipos básicos para adequá-los a necessidades específicas
- Modificadores:
  - **signed**: indica número com sinal (inteiros e caracteres)
  - **unsigned**: número apenas positivo (inteiros e caracteres)
  - **long**: aumenta abrangência (inteiros e reais)
  - **short**: reduz a abrangência (inteiros)

Abrangência de dados: 16 bits

Tipo	Tamanho (bytes)	Abrangência	
char	1	−128	a 127
unsigned char	1	0	a 255
int	2	−32768	a 32767
unsigned int	2	0	a 65535
short int	2	−32768	a 32767
long int	4	−2.147.483.648	a 2.147.483.647
unsigned long int	4	0	a 4.294.967.295
float	4	$\pm 3{,}4 \cdot 10^{-38}$	a $\pm 3{,}4 \cdot 10^{38}$
double	8	$\pm 1{,}7 \cdot 10^{-308}$	a $\pm 1{,}7 \cdot 10^{308}$
long double	10	$\pm 3{,}4 \cdot 10^{-4932}$	a $\pm 3{,}4 \cdot 10^{4932}$

Abrangência de dados: 32 bits

Tipo	Tamanho (bytes)	Abrangência
char	1	−128 a 127
unsigned char	1	0 a 255
int	4	−2.147.483.648 a 2.147.483.647
unsigned int	4	0 a 4.294.967.295
short int	2	−32768 a 32767
long int	4	−2.147.483.648 a 2.147.483.647
unsigned long int	4	0 a 4.294.967.295
float	4	± 3,4·10 $^{-38}$ a ± 3,4·10 ^{38}
double	8	± 1,7·10 $^{-308}$ a ± 1,7·10 ^{308}
long double	10	± 3,4·10 $^{-4932}$ a ± 3,4·10 ^{4932}

Declaração de variáveis

- A declaração de uma variável segue o modelo:

  TIPO_VARIÁVEL lista_de_variaveis;

- Ex:

```
int x, y, z;
float f;
unsigned int u;
long double df;
char c = 'A';          /* variavel definida e iniciada */
char s[ ] = "vetor de caracteres";
```


Programação em C

OPERADORES

Operadores

- Correspondem a símbolos simples ou combinados que representam operações de natureza: aritmética, relacional ou lógica.
- Podem ser classificados também quanto a quantidade de elementos sob os quais incidem, i.e., unários, binários ou ternários

Op. Aritméticos

- Representam as operações aritméticas básicas

Operação	Operador
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Resto	%
Incremento	++
Decremento	--

Op. Relacionais

- Estabelecem relações/comparações

Operação	Operador
Igualdade	==
Diferença	!=
Maior	>
Maior ou igual	>=
Menor	<
Menor ou igual	<=

Op. Lógicos

- Representam as operações básica dada na lógica matemática

Operação	Operador
Negação	!
Conjunção	&&
Disjunção	||

Op. de Atribuição (de variável)

- Forma geral:

  variavel = expressão ou constante

- Armazena o conteúdo dado a direita no elemento dado à esquerda
- Múltiplas atribuições
  - C permite a atribuição de mais de uma variável em um mesmo comando:

```
x = y = z = 0;
```


Expressões

- Expressões são compostas por:
  - Operandos: a, b, x, Meu_dado, 2, ...
  - Operadores: +, -, %, ...
  - Pontuação: ( )
  - Funções: sin(), abs(), sqrt(), ...
- Ex:

```
X
14
x + y
(x + y)*z + w – v
( -b + sqrt(delta) ) / 2*a
```


Expressões

- Expressões retornam um valor:

```
x = 5 + 4 /* retorna 9 */
```

  - esta expressão atribui 9 a x e retorna 9 como resultado da expressão

```
((x = 5 + 4) == 9) /* retorna verdade = 1 */
```

  - na expressão acima, além de atribuir 9 a x, o valor retornado é utilizado em uma comparação

Expressões

- a ordem em que uma expressão é avaliada depende da prioridade dos operadores e da pontuação
- expressões podem aparecer em diversos pontos de um programa
  - comandos — /* x = y; */
  - parâmetros de funções — /* sqrt(x + y); */
  - condições de teste — /* if (x == y) */

Conversão de Tipos

- Quando uma variável de um tipo é atribuída a uma de outro tipo, o compilador automaticamente converte o tipo da variável a direita de "=" para o tipo da variável a esquerda de "="
- Ex:

```
int i; char ch; float f;
ch = i;     /* ch recebe 8 bits menos significativos de x */
i = f;      /* x recebe parte inteira de f */
f = ch;     /* f recebe valor 8 bits convertido para real */
f = i;      /* idem para inteiro i */
```

Programação em C

ESTRUTURA BÁSICA

Programação em C

- Todo programa, escrito na linguagem C, deve apresentar uma função principal chamada main, que define todo o corpo do programa
- Exemplo:

```
int main() {
  /* corpo do programa */
}
```

Comandos de Saída

- Empregados para que o sistema forneça, em um dispositivo de saída, as mensagens e resultados de seu processamento.
- O dispositivo padrão de saída é o monitor.
- A linguagem C oferece alguns comandos de saída, mas o que apresenta propósito mais geral é o printf.

Comando PRINTF()

- Sintaxe:

  printf("Mensagem", lista de variáveis);

- Funcionamento:
  - O comando escreve a mensagem dada no dispositivo padrão de saída, realizando a substituição das máscaras de formatação encontradas pelas respectivas variáveis dadas na lista subseqüente a mensagem.
  - O dispositivo padrão é dado pela variável stdout

Máscaras de formatação

- Símbolo de por cento seguido de uma letra:

- %c Caractere
- %d Inteiros com sinal
- %u Inteiros sem sinal
- %f Números reais
- %e Notação científica
- %x Números em hexadecimal
- %s Cadeia de caracteres (strings)

Exemplo

- Saída formatada printf(). Ex:
  - O trecho abaixo:

    int i = 10;

    float r = 3.1514;

    char s[] = "Blablabla"; /* cadeia de caracteres */

    printf("Inteiro: %d, Real: %f, String: %s",i,r,s);
  - Produz:

    Inteiro: 10, Real: 3.151400, String: Blablabla

Constantes do Tipo Char

- Barra invertida seguido de um caractere:

- \a bip
- \b backspace
- \n nova linha
- \r return
- \t tabulação horizontal
- \’ apóstrofe
- \” aspas
- \\ barra invertida

Comandos de entrada

- Utilizado para receber dados fornecidos pelo usuário (dados de entrada) e armazená-los na memória principal (em variáveis)
- Os dados são fornecidos ao sistema por meio de um dispositivo de entrada, cuja configuração dada como padrão é o teclado.
- A linguagem C oferece vários comandos de entrada, cada qual mais indicado para uma situação em particular.
- O principal comando de entrada é o scanf

Comando SCANF()

- Sintaxe:

  scanf(“formato”, &variável);

- Funcionamento:
  - O comando coleta as informações dadas no dispositivo padrão de entrada, interpretando as informações segundo a máscara de formatação e armazenando na(s) respectiva(s) variável(is) dada(s) subseqüentemente ao formato.
  - O dispositivo padrão é dado pela variável stdin

Exemplo

- Entrada formatada scanf().
  - Ex:

    int i; **float** r; **char** str[10];
    scanf(“%d”,&i);
    scanf(“%f”,&r);
    scanf(“%s”,&str);
  - Ou ainda:

    int dia, mes, ano;
    scanf(“%d/%d/%d”, &dia, &mes, &ano);

Programação em C

UM EXEMPLO COMPLETO

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Informa ao pré-processador os arquivos de funções (bibliotecas) que precisam ser vinculados a esse programa para a construção do arquivo executável

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Função Principal;
Característica fundamental de todo o programa C

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Bloco de Comandos:
Define as ação/instruções que serão executadas;
Tem seu início e fim delimitado por chaves

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

raio, **area** e **perim** são identificadores e precisam ser declarados antes de serem utilizados.

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Comando de saída:
Escreve a mensagem "Forneça o valor do RAIO:" na tela

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Comando de entrada:
Realiza a leitura do teclado, interpretando a informação como um número inteiro e armazenando na variável raio

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Operador de Atribuição: Armazena o resultado da expressão na variável dada a esquerda, isto é, na variável area

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Operador de Atribuição: Armazena o resultado da expressão na variável perim

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Constantes: Dadas por identificadores, representam valores específicos. Neste caso, equivale ao valor de π e está definida na biblioteca MATH.H

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Comando de saída: Escreve na tela a mensagem e o valor armazenado na variável area, saltando para a próxima linha ao término

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Comando de saída: Escreve na tela a mensagem e o valor armazenado na variável perim, saltando para a próxima linha ao término

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Comando de saída: Salta para a próxima linha e escreve a mensagem "Pressione qq tecla para continuar ..."

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Comando de entrada:
Aguarda até que o usuário pressione alguma tecla, retornando seu valor para o programa (que para nesse caso não está sendo armazenado na memória)

Programa C

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

int main(){
    int raio;
    float area, perim;
    printf("Forneca o valor do RAIO: ");
    scanf("%d",&raio);
    area = M_PI * raio * raio;
    perim = 2 * M_PI * raio;
    printf("Area: %f\n",area);
    printf("Perimetro: %f\n",perim);
    printf("\nPressione qq tecla para retornar ...");
    getch();
    return 0;
}
```

Retorna o valor 0 ao Sistema Operacional, como um código indicando sucesso na execução do programa