FAETERJ - Paracambi
Análise e Desenvolvimento de Sistemas
Programação Estruturada – PRG - I

Prof. Carlos Eduardo Costa Vieira

Resumo da Apresentação

- A Função `main()` e as Bibliotecas;
- Declaração de Variáveis e Constantes;
- Tipos de Dados;
- Entrada e Saída de Dados;
- Comando de Atribuição;
- Operadores e Comentários;
- Estrutura Sequencial;
- Bibliografia.

A Função `main()`

- Função essencial para execução de programas em C;
- Presente nos códigos;
- Primeira função chamada quando a execução de um programa inicia;
- Não pode ser usado como nome de variável.

Bibliotecas

- São arquivos contendo várias funções que podem ser incorporadas aos programas escritos em C;
- A diretiva `#include` faz o texto contido na biblioteca especificada ser inserido no programa;
- Exemplo:
  - `#include <stdio.h>`
  - A biblioteca `stdio.h` permite a utilização de diversos comandos de entrada e saída.

Ex: Função `main()` e Bibliotecas

- Exemplo *(Aula1.1.c)*

```
#include <stdio.h>
int main(){
   printf("************************\n");
   printf("Bem vindo ao curso de PE.\n");
   printf("************************\n");
   return 0;
}
```

Variáveis

- Dados cujos valores variam durante a execução do programa;
- São entidades que representam dados do programa e possuem nome e valor;
- Representam uma posição de memória do computador em um dado momento. Posteriormente, todos os valores podem ser alterados;
- Detalhes importantes em uma variável: o identificador da variável e o tipo de valor que essa variável irá conter.

Declaração de Variáveis

- Exemplo

```
#include <stdio.h>
int main (){
//Declaração de variáveis
char sexo;
char pedido1[30], pedido2[30];
float val1, val2, val3,quant1,quant2;
int n1,n2,n3;
double numero;
…}
```

Tipos de Dados

Tipo	Tamanho (bits)	Intervalo
char	8	-128 a 127
int	16	-32768 a 32767
float	32	3.4E-38 a 3.4E+38
double	64	1.7E-308 a 1.7E+308
void	0	sem valor
unsigned char	8	0 a 255
unsigned int	16	0 a 65535
long int	32	-2147483648 a 2147483647
unsigned long int	32	0 a 4294967295
long double	80	3.4E-4932 a 1.1E+4932

Regras para a Declaração de Variáveis

- O nome de uma variável pode ser constituído por letras do alfabeto (minúsculas ou maiúsculas), dígitos (0,1, … ,9) e ainda pelo caracter *underscore* (_);
- O primeiro caracter não pode ser um dígito. Terá que ser uma letra ou caractere underscore (mas é desaconselhável);
- Maiúsculas e minúsculas representam caracteres diferentes, logo variáveis distintas (a linguagem é *Case Sensitive*);
- Uma variável não pode ter por nome uma palavra reservada da linguagem;
- O nome deve refletir o significado da variável.

Declaração de Constantes

- São declaradas depois das bibliotecas e seus valores não podem ser alterados durante a execução do programa;
- Sintaxe:
  - `#define NOME Valor_Const`
  - `Exemplos:`
    - `#define TAM 10`
    - `#define X 100`

Entrada de Dados

- Comando de Entrada
- Sintaxe
  - `scanf("<tipo de dado>",&<variável>);`
    - Comando nativo da linguagem C;
    - É obrigatório a utilização da biblioteca `stdio.h`;
    - O operador & é utilizado para obter o endereço de memória da variável;
    - Ex: `scanf("%d",&a);`

Saída de Dados

- Comando de Saída
- Sintaxe
  - `printf("<tipo de dado>",<variável>);`
    - Comando nativo da linguagem C;
    - É obrigatório a utilização da biblioteca `stdio.h`;
    - Ex: `printf("Valor de: %d",a);`

Impressão dos Tipos de Dados

Código	Significado
%c	Exibe um caractere
%s	Exibe uma string (conjunto de caracteres)
%d ou %i	Exibe um inteiro
%f	Exibe um ponto flutuante
%lf	Exibe um ponto flutuante com dupla precisão
%e	Exibe um número em notação científica (e minúsculo)
%E	Exibe um número em notação científica (E maiúsculo)
%o	Exibe um número em notação octal
%x	Exibe um número em hexadecimal (letras minúsculas)
%X	Exibe um número em hexadecimal (letras maiúsculas)

Impressão de Códigos Especiais

Código	Significado
\n	Leva o cursor para a próxima linha
\t	Executa uma tabulação
\b	Executa um retrocesso
\f	Leva o cursor para a próxima página
\”	Exibe o caractere aspas (”)
\'	Exibe o caractere apóstrofo (’)
\\	Exibe o caractere barra invertida (\)
%%	Exibe o caractere %

Entrada e Saída de Dados

- Exemplo *(Aula1.2.c)*

```
#include <stdio.h>
int main(){
    float a,g;
    printf("Digite o preco do alcool: ");
    scanf("%f",&a);
    printf("Digite o preco da gasolina: ");
    scanf("%f",&g);
    printf("Preco da gasolina: %f Preco do
    alcool: %f.\n",g,a);
    return 0;
}
```

Saída de Dados

- Formatação *(Aula1.3.c)*

```
#include <stdio.h>
int main(){
    printf("Um caracter impresso: %c\n”,'x');
    printf("Uma string impressa: %s\n","Entendi tudo.");
    printf("Número impresso: %f\n",3.141517);
    printf("Uma Casa: %.1f\n",3.141517);
    printf("Duas Casas: %.2f\n",3.141517);
    printf("Tres Casas: %.3f\n",3.141517);
    printf("Notacao Cientifica: %e\n",3.141517);
    printf("Valor: %5d \n",25);
    printf("Valor: %10d \n",25);
    return 0;
}
```

Comando de Atribuição

- A instrução de atribuição permite que o conteúdo de uma variável seja alterado (operador =);
- Sintaxe:
  - `Nome_Var = valor_var;`
  - Exemplos:

```
int a,b,c,d;
a = 5;
b = 7;
c = a;
d = a + b + c;
c = c + b;
c = c * a;
```

Operadores Aritméticos

Operador	Operação Matemática	Prioridade
+	Adição	5
-	Subtração	5
%	Resto da divisão	4
*	Multiplicação	3
/	Divisão	3
++	Incremento	2
--	Decremento	2
+	Manutenção do sinal	1
-	Inversão do sinal	1

Op. Aritméticos: Exemplo (*Aula1.4.c*)

```
#include <stdio.h>
int main(){
    int n1,n2;
    printf("Digite o primeiro número: ");
    scanf("%d",&n1);
    printf("Digite o segundo número: ");
    scanf("%d",&n2);
    printf("\nA operação %d + %d = %d.\n",n1,n2,n1+n2);
    printf("A operação %d - %d = %d.\n",n1,n2,n1-n2);
    printf("A operação %d * %d = %d.\n",n1,n2,n1*n2);
    printf("A operação %d / %d = %d.\n",n1,n2,n1/n2);
    printf("A operação %d %% %d = %d.\n",n1,n2,n1%n2);
    n1++;
    n2--;
    printf("Incremento de n1: %d.\n",n1);
    printf("Decremento de n2: %d\n\n",n2);
}
```

Operadores Relacionais

Operadores Relacionais	C
Maior	>
Menor	<
Maior ou igual	>=
Menor ou igual	<=
Igual	==
Diferente	!=

Operadores Lógicos

- Tabelas-Verdade (0 – falso; 1 – verdadeiro)

A	B	A && B	A || B	!A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Op. Relac. e Lógicos : Ex. (*Aula1.5.c*)

```
#include <stdio.h>
int main(){
   int n1,n2;
   printf("Digite o primeiro número: ");
   scanf("%d",&n1);
   printf("Digite o segundo número: ");
   scanf("%d",&n2);
   printf("\nA op. %d > %d é igual a: %d.\n",n1,n2,n1>n2);
   printf("A op. %d < %d é igual a: %d.\n",n1,n2,n1<n2);
   printf("A op. %d = %d é igual a: %d.\n",n1,n2,n1==n2);
   printf("A op. %d <> %d é igual a: %d.\n",n1,n2,n1!=n2);
   printf("A op. (%d>%d) E (%d=%d) é igual a: %d.
   \n",n1,n2,n1,n2,(n1 > n2) && (n1 == n2));
   printf("A op. (%d>%d) OU (%d<>%d) é igual a: %d.
   \n\n",n1,n2,n1,n2,(n1 > n2) || (n1 != n2));
   return 0;
}
```

Comentários

- Declarações não compiladas que podem conter qualquer informação textual que você queira adicionar ao código-fonte para referência e documentação de seu programa;
- Uma linha (utilizar //)
  - `// linha de comentario`
- Várias linhas (utilizar /* ..... */)
  - `/*`
  - `  linha de comentario`
  - `  linha de comentario`
  - `*/`

Estrutura Sequencial

- Em uma Estrutura Sequencial, os comandos do algoritmo são executados em uma sequência linear (de cima para baixo, um após o outro e uma única vez), sem que haja desvios ou repetições na sequência das instruções;

```
#include <nome_da_biblioteca>
int main(){
   bloco de comandos;
   return 0;
}
```

Exemplo (*Aula1.6.c*)

```
#include <stdio.h>
int main()
{
   int n1,n2,soma,dif;
   printf("Digite o 1° valor: ");
   scanf("%d",&n1);
   printf("Digite o 2° valor: ");
   scanf("%d",&n2);
   soma = n1+n2;
   dif = n1-n2;
   printf("\nSoma: %d + %d = %d.\n",n1,n2,soma);
   printf("Subtração: %d - %d =
%d.\n\n",n1,n2,dif);
   return 0;
}
```

Bibliografia

- ASCENCIO, A. F. G.; CAMPOS, E. A. V. de. **Fundamentos da Programação de Computadores:** Algoritmos: Pascal, C/C++ e Java. 3. ed. São Paulo: Pearson Prentice Hall, 2012.
- DEITEL, P. J; DEITEL, H. M. **C:** Como Programar. 6. ed. São Paulo: Pearson Prentice Hall,2011.
- OLIVEIRA, Ulysses. **Programando em C:** Fundamentos. 22 ed. Rio de Janeiro: Ciência Moderna, 2008.