

**USP - ICMC - SSC
SSC 0501 - 1o. Semestre 2010**

**Disciplina de
Introdução à Ciência da Computação
ICC 1 - Teoria**

Prof. Fernando Santos Osório

Email: fosorio [at] { icmc. usp. br , gmail. com }

Página Pessoal: <http://www.icmc.usp.br/~fosorio/>

Web - WIKI ICMC: <http://wiki.icmc.usp.br/index.php/SSC-501>

PAE: Gustavo Pessin (Dout. CCMC) / <Http://pessin.googlepages.com/>

Email: pessin [at] { icmc.usp.br , gmail.com }

Monitor: Matheus Lin Alvarenga (EC) / <Http://matheuslin.wordpress.com/>

Email: matheus.lin [at] gmail.com

Aula 02s

1

Março 2010

Aula 02 - Programação em "C"

Agenda:

1. Algoritmo => Fluxograma => Codificação (Código do Programa)
2. Linguagem de Programação "C"
3. Projeto, Codificação, Compilação, Execução e Teste
4. Programa: Instruções e Dados (Memória)
5. Tipos de Dados
6. Declaração de Variáveis
7. Expressões
8. Entrada e Saída de Dados (printf / scanf)
9. Programas Seqüenciais Simples

2

Março 2010

- Estrutura seqüencial, os passos são definidos e executados em uma seqüência pré-definida.



Estrutura Seqüencial



```
#include <stdio.h>
#include <stdlib.h>
char Nome[30];
int main(int argc, char *argv[])
{
    /* 1: Perguntar Nome */
    printf("Qual o seu nome?\n");
    /* 2: Ler o Nome */
    scanf("%s", Nome);
    /* 3: Escrever "Hello" */
    printf("Hello World! Hello...");
    /* 4: Escrever Nome */
    printf("%s\n", Nome);
    /* 5: Escrever "Tchau" */
    printf("\nTchau!\n\n");
    /* 6: FIM */
    system("PAUSE");
    return 0;
}
```

Como passar de um Algoritmo Para um Programa de Computador

Computador:

- Uso de dados armazenados na *memória* (variáveis)
- Instruções bem definidas: os *comandos* da linguagem

Ciclo:

- **Entrada de Dados:** Ler os dados
- **Processamento:** manipular os dados
- **Saída de Dados:** Escrever os resultados

Programa de Computador: Memória

1	2	3	4	5	6	7	...		

A memória do computador armazena dados (bytes)
Cada dado tem a sua posição na memória (endereço)

Programa de Computador: Memória

1	2	3	4	5	6	7	...		
Pregos	Porcas	Parafusos	Açúcar	Sal	Óleo	Leite			

A memória do computador armazena dados (bytes)
Cada endereço pode armazenar diferentes tipos de dados (variáveis)

Programa de Computador: Memória

1	2	3	4	5	6	7	...		
Pregos	Porcas	Parafusos	Açúcar	Sal	Óleo	Leite			
30	45	45	2kg	300g	1L	250ml			

A memória do computador armazena dados (bytes)
Cada variável armazena uma informação (valor da variável)

7

Março 2010

Programa de Computador: Memória

1	2	3	4	5	6	7	...		
Pregos	Porcas	Parafusos	Açúcar	Sal	Óleo	Leite			
30	45	45	2kg	300g	1L	250ml			

Quantidade_de_Pregos_Disponiveis

Variáveis vão receber NOMES...
Nomes que representam uma informação (valor armazenado)
de um determinado tipo em uma determinada posição da memória

8

Março 2010

Programa de Computador: Comandos

Comandos são ordens para que o computador manipule os dados de sua memória...

Exemplos de Comandos:

- Realizar operações com os dados: mover, somar, subtrair, ...
- Ler novos dados pelo teclado: entrada de dados
- Escrever resultados na tela: saída de dados

Programa de Computador: Comandos

Comandos são ordens para que o computador manipule os dados de sua memória...

Exemplos de Comandos:

- Realizar operações com os dados: mover, somar, subtrair, ...
- Ler novos dados pelo teclado: entrada de dados
- Escrever resultados na tela: saída de dados

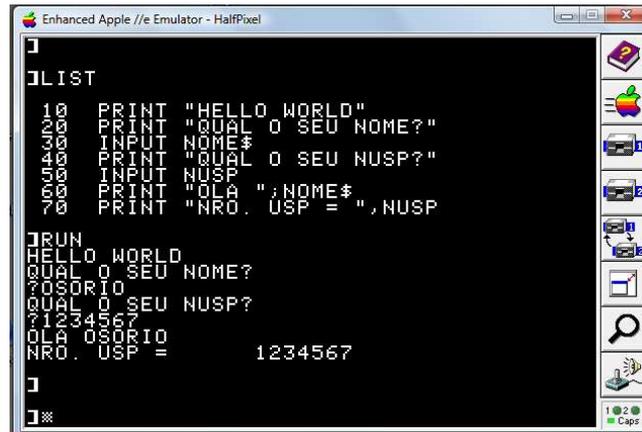
ESCOLHA DA LINGUAGEM PROGRAMAÇÃO:

http://pt.wikipedia.org/wiki/Linguagem_de_programação

http://pt.wikipedia.org/wiki/Anexo:Lista_de_linguagens_de_programação

Programa de Computador: Comandos

Comandos são ordens para que o computador manipule os dados de sua memória... Exemplo de Programa: BASIC!



```
Enhanced Apple //e Emulator - HalfPixel
]
]LIST
10 PRINT "HELLO WORLD"
20 PRINT "QUAL O SEU NOME?"
30 INPUT NOME$
40 PRINT "QUAL O SEU NUSP?"
50 INPUT NUSP$
60 PRINT "OLA " ; NOME$
70 PRINT "NRO. USP = ", NUSP

]RUN
HELLO WORLD
QUAL O SEU NOME?
?OSORIO
QUAL O SEU NUSP?
?1234567
OLA OSORIO
NRO. USP =      1234567
]
]~
```

Programa de Computador: Comandos

Comandos são ordens para que o computador manipule os dados de sua memória...

Exemplo de Programa: LINGUAGEM "C"

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

int main(int argc, char *argv[])
{
    printf("Hello World\n");
    system("PAUSE");
    return 0;
}
```

Programa de Computador: Comandos

```
/* Comentário: Este é um Exemplo de Programa em "C" */  
  
#include <stdio.h>  
#include <stdlib.h>  
char Nome[30];  
  
int main(int argc, char *argv[])  
{  
    printf("Qual o seu nome? ");  
    scanf ("%s",Nome);  
    printf("Hello %s\n",Nome);  
    system("PAUSE");  
    return 0;  
}
```

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 \neq nome \neq NOME
- Não pode ser empregar qualquer uma das palavras reservadas à linguagem C como identificadores

Palavras-chave de C (ANSI)

Palavras Reservadas

```
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

Contador
Teste23
Alto_Paraíso
__sizeint

Incorretos

1contador
oi!gente
Alto..Paraíso
_size-int

Variáveis

- Escolha a opção que inclui somente nomes válidos para variáveis na linguagem C:

- a) if, a_b_2, H789, _yes
- b) i, j, int, obs
- c) 9xy, a36, x*y, --j
- d) 2_ou_1, \fim, *h, j
- e) Nenhuma das opções anteriores

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* ou *double* 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 e do compilador usado, tipicamente 16 ou 32 bits (2 ou 4 bytes)
- *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 a precisão (inteiros e reais)
 - **short**: reduz a precisão (inteiros e reais)

Precisão dos dados

Tipo	Tamanho (bytes)	Abrangência dos Valores	
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 $\pm 3,4 \cdot 10^{4932}$

Precisão dos dados

Tipo	Tamanho (bytes)	Abrangência dos Valores	
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	$\pm 3,4 \cdot 10^{-38}$	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 $\pm 3,4 \cdot 10^{4932}$

A precisão depende do compilador e da máquina usada...

Precisão dos dados

Porque o tipo e a precisão dos dados é tão importante?
Pergunte aos engenheiros do projeto do ARIANE 5...

YouTube
ariane explosion
Pesquisar

Página inicial Vídeos Canais

Wikipedia:
http://en.wikipedia.org/wiki/Ariane_5_Flight_501

Ariane 5 Rocket Explosion

0:07 / 0:18

<http://www.youtube.com/watch?v=z-r9cYp3tTE>

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[20] = "vetor de caracteres";
```

Linguagem "C"

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

Exemplo: unário `Valor++`

binário `Valor1 + Valor2`

Operadores Aritméticos

- Representam as operações aritméticas básicas

Operação	Operador
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Resto da Divisão	%
Incremento (+1)	++
Decremento (-1)	--
Sinal Negativo	-

Operadores Relacionais

- Estabelecem relações/comparações

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

Operadores Lógicos

- Representam as operações básica dada na lógica matemática

Operação	Operador
Negação	!
Conjunção (E)	&&
Disjunção (OU)	

Operadores de Atribuição

- Forma geral:
 $variavel = expressão\ ou\ constante$
- Armazena o conteúdo dado a direita no elemento dado à esquerda
 $Salario_Minimo = 465.00;$
- 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 da biblioteca do "C": sin(), abs(), sqrt(), ...
- Exemplos:
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 */
```

Linguagem “C”

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 */  
}
```

39

Março 2010

Programação em C

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

```
#include <stdio.h>  
#include <stdlib.h>  
  
int main(int argc, char *argv[])  
{  
    /* corpo do programa */  
    system ("PAUSE");  
    return 0;  
}
```

40

Março 2010

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 subsequente 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 (float)

`%lf` Números reais (double ou long float)

`%s` Cadeia de caracteres (strings)

`%e` Notação científica

`%x` Números em hexadecimal

Exemplo

- Saída formatada PRINTF().

Exemplo:

– O trecho abaixo:

```
int i = 10;
```

```
float r = 3.1514;
```

```
char s[10] = "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
`\t` tabulação horizontal
`\'` apóstrofe
`\"` aspas
`\\` barra invertida
`\f` form feed

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().

- Exemplos:

```
int idade; float salario; double x; char nome[10];
```

```
scanf("%d",&idade);
```

```
scanf("%f",&salario);
```

```
scanf("%lf",&x);
```

```
scanf("%s",nome);
```

- Ou ainda:

```
int dia, mes, ano;
```

```
scanf("%d/%d/%d", &dia, &mes, &ano);
```

Programação em C

UM EXEMPLO COMPLETO

Programa C

```
/* Programa: calculo da área e do perímetro
de uma circunferência */

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

int main()
{
    int raio;
    float area, perim;

    printf("Forneca o valor do RAIIO: ");
    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;
}
```



INFORMAÇÕES SOBRE A DISCIPLINA

USP - Universidade de São Paulo - São Carlos, SP
ICMC - Instituto de Ciências Matemáticas e de Computação
SSC - Departamento de Sistemas de Computação

Prof. Fernando Santos OSÓRIO

Web institucional: <http://www.icmc.usp.br/ssc/>

Página pessoal: <http://www.icmc.usp.br/~fosorio/>

E-mail: [fosorio \[at\] icmc. usp. br](mailto:fosorio@icmc.usp.br) ou [fosorio \[at\] gmail. com](mailto:fosorio@gmail.com)

PAE Gustavo Pessin – E-mail: [pessin \[at\] gmail .com](mailto:pessin@gmail.com)

Monitor Matheus Lin – E-mail: [matheus.lin \[at\] gmail.com](mailto:matheus.lin@gmail.com)

Disciplina de Introdução a Ciência da Computação

Web disciplina: Wiki ICMC - [Http://wiki.icmc.usp.br](http://wiki.icmc.usp.br)

> Programa, Material de Aulas, Critérios de Avaliação,

> Trabalhos Práticos, Datas das Provas, Notas