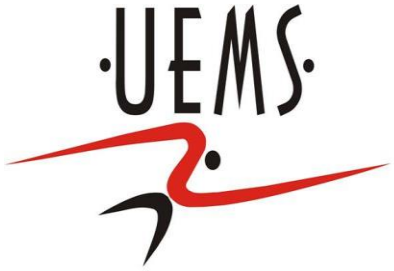


PCI- Variáveis, Atribuição, I/O

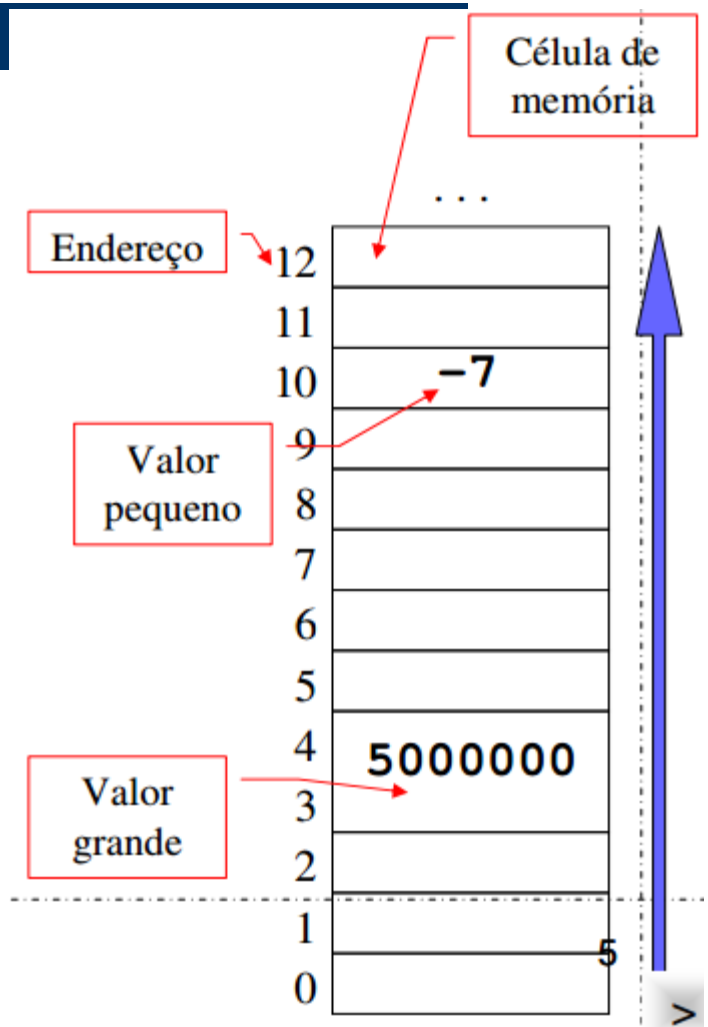
Profa. Mercedes Gonzales Márquez

Memória

- **Memória:** sequência de células
- **Endereço:** posição da célula
- **Células armazenam dados**

– **Valor pequeno:** uma célula

– **Valor grande:** duas ou mais células



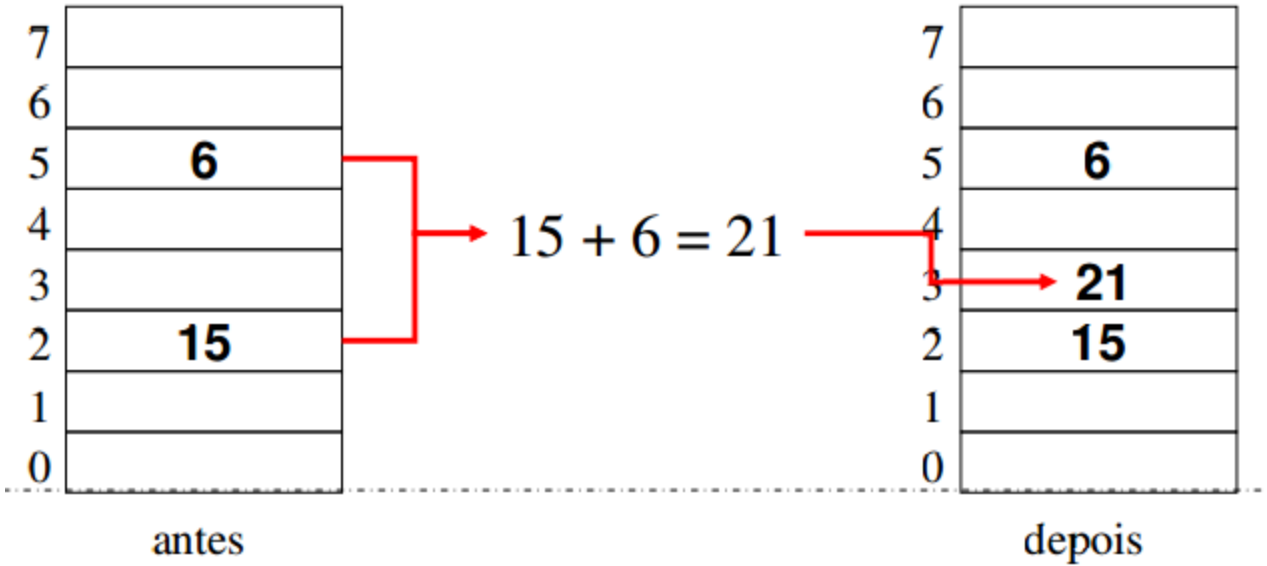
Memória

- **Operações na memória:**

1. Consulta (lê) células de memória

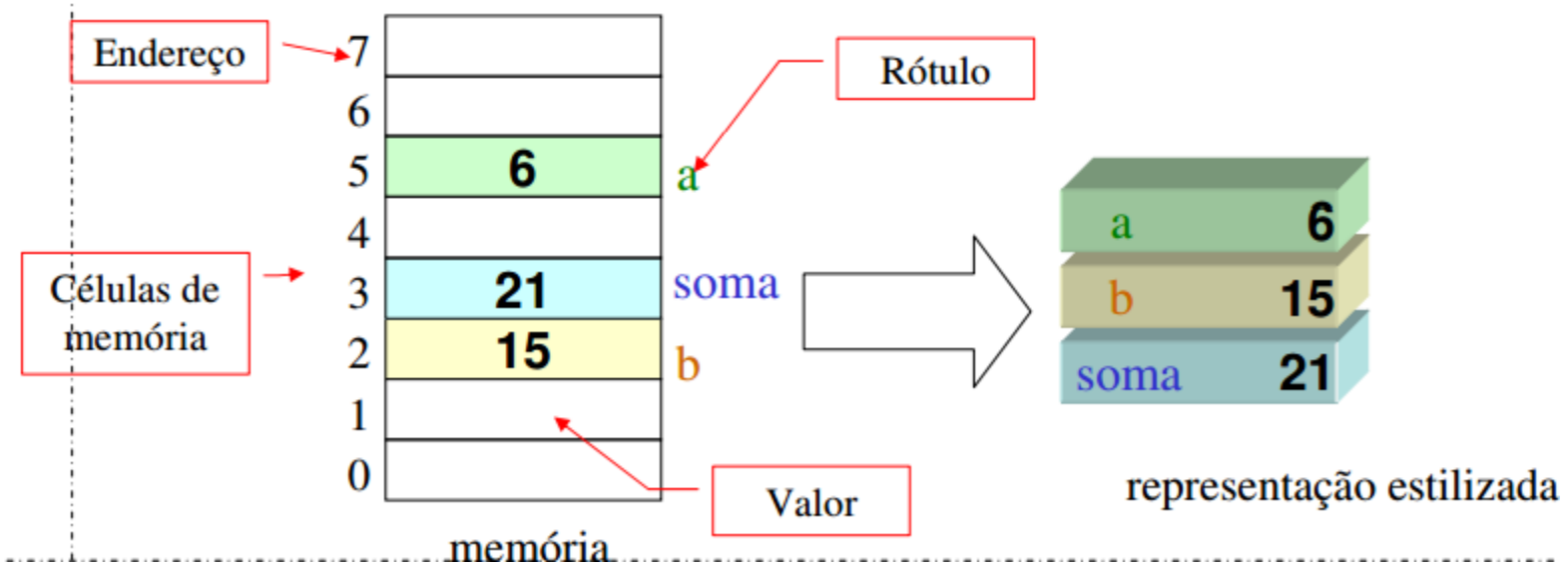
2. Programa calcula um novo valor

3. Armazena (escreve) o novo valor em uma célula



Variáveis

- Abstração dos endereços de memória (posição nomeada de memória)
- Rótulo simbólico para cada endereço.



Declarando uma variável

- Todas as variáveis devem ser declaradas antes de serem utilizadas.
- Declara-se da seguinte forma:

  Tipo Variável Nome Variável;

  Exemplo :

  int numero;

Variáveis inteiras

Os seguintes são os tipos da linguagem C que servem para armazenar inteiros:

int: Inteiro cujo comprimento depende do processador. E o inteiro mais utilizado. Em processadores Intel comum, ocupa 32 bits e pode armazenar valores de -2.147.483.648 a 2.147.483.647.

unsigned int: Inteiro cujo comprimento depende do processador e que armazena somente valores positivos. Em processadores Intel comum, ocupa 32 bits e pode armazenar valores de 0 a 4.294.967.295.

Variáveis inteiras

long int: Inteiro que ocupa 64 bits em computadores Intel de 64bits e pode armazenar valores de aprox. -9 $x10^{18}$ a aprox. 9 $x10^{18}$.
unsigned long int: Inteiro que ocupa 64 bits e em computadores Intel de 64bits e armazena valores de 0 ate aprox. 18 $x10^{18}$.
short int: Inteiro que ocupa 16 bits e pode armazenar valores de -32.768 a 32.767.
unsigned short int: Inteiro que ocupa 16 bits e pode armazenar valores de 0 a 65.535.

Variáveis de tipo caractere

Variáveis utilizadas para armazenar letras e outros símbolos existentes em textos. Guarda apenas um caractere.
Exemplos de declaração:
char umaLetra;

Variáveis de tipo ponto flutuante

Armazenam valores reais. Mas possuem problemas de precisão pois há uma quantidade limitada de memória para armazenar um número real.
Exemplos de números em ponto flutuante: 2.1345 ou 9098.123.

- **float:** Utiliza 32 bits, e na pratica tem precisão de aproximadamente 6 casas decimais (depois do ponto). Pode armazenar valores de $(+/-)10^{-38}$ a $(+/-)10^{38}$

double: Utiliza 64 bits, e na pratica tem precisão de aproximadamente 15 casas decimais. Pode armazenar valores de $(+/-)10^{-308}$ a $(+/-)10^{308}$

Regras para nomes de variáveis em C

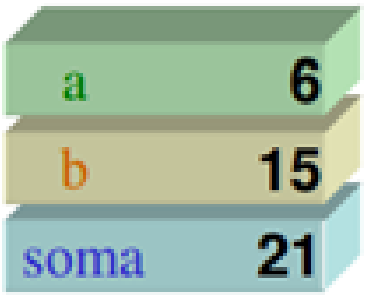
- Deve começar e pode conter letra (maiúscula ou minúscula) ou underline(_).
- Não pode-se utilizar como parte do nome de uma variável: { (+ - * / \ ; . , ?
- Letras maiúsculas e minúsculas são diferentes:
- As seguintes palavras são reservadas na linguagem C e portanto não podem ser utilizadas como nome de variáveis:

auto double int struct break enum register typedef char extern return union const float short unsigned continue for signed void default goto sizeof volatile do

if static while

Comando de atribuição

Sintaxe:
variável = valor; ou
variável=expressão




```
int main{
    int a,b,soma;
    a = 6; // o valor 6 atribui-se para a
    b=15; // o valor 15 atribui-se para b
    soma=a+b;// uma expressão é
atribuída para soma
}
```

Comandos de entrada/saída

Printf para saída de dados:
O comando printf apresenta números e caracteres na tela.
Sintaxe: printf (*string de controle* , *argumento1, argumento2, ..., argumento n*)

Uma string de controle pode ter :

- sequência de conversão (especifica um formato, inicia com %)
- Sequência de escape (inicia com barra invertida \)

Comandos de entrada/saída

Exemplo de impressão de uma string de controle sem sequência de conversão.

```
int main(){
    printf (“Imprimindo Notas”);
}
```

As principais sequências de conversão são:

%c imprime o conteúdo da variável com representação ASCII;

%d imprime o conteúdo da variável com representação decimal com sinal;

%f imprime o conteúdo da variável com representação com ponto decimal;

%u imprime o conteúdo da variável com representação decimal sem sinal;

Comandos de entrada/saída

Exemplo de uso de um único argumento com sequência de conversão: Se x é uma variável inteira com valor 12.
printf("Valor de x = %d", x);
imprime na tela a frase Valor de x = 12.

Exemplo do uso de 2 argumentos também com sequência de conversão:
Se y é uma variável do tipo char com valor 'A', então a execução de
printf("x = %d e y = %c", x, y);
imprime na tela a frase x = 12 e y = A

Comandos de entrada/saída

Sequências de escape mais utilizadas:

Sequência	Significado
\n	Quebra de linha (*line feed* ou LF)
\t	Tabulação horizontal
\b	Retrocede o cursor em um caractere (backspace)
\r	Retorno de carro (*carriage return* ou CR): volta o cursor para o começo da linha sem mudar de linha
\a	Emite um sinal sonoro
\"	Aspa dupla
\'	Aspa simples
\\	Barra invertida
\0	Caractere nulo (caractere de valor zero, usado como terminador de strings)

Comandos de entrada/saída

Exemplo usando sequências de escape

```
#include<stdio.h>
#define PRECO 1.99
int main(){
   int pera = 3;
   char qualidade = ’A’;
   float peso = 2.5, total=peso*PRECO;
   printf(“Existem %d peras de qualidade %c “, pera, qualidade);
   printf(“pesando %f quilos.\n”, peso);
   printf(“O preco por quilo e %f, total e %f\n”, PRECO, total);
}
```

Comandos de entrada/saída

Observações:

- Uma largura de campo pode ser opcionalmente especificada logo após o caráter %, como em %3d onde o número decimal terá reservado um espaço de três caracteres para sua representação.
- Em casos de números reais por exemplo %12.3f especifica que a variável será apresentada em um campo de doze caracteres com uma precisão de três dígitos após o ponto decimal. Exemplo:

```
int main(){
    int a=9, b=800;
    float c=20.9176, d=-1.4219;
    printf (“\n a=%3d,\n b=%3d, \n c=%5.2f,d=%5.2f\n”,a,b,c,d);
}
```

Comandos de entrada/saída

Entrada de Dados:
O teclado é o dispositivo padrão para entrada de dados. O comando scanf lê números e caracteres digitados no teclado da seguinte forma:
scanf (“%d”, &x) onde x é uma variável inteira e &x é o endereço em memória da variável x.
Cada tipo de variável requer um símbolo específico de conversão.
O símbolo %d é usado para variáveis int,short, unsigned short, unsigned char;
O símbolo %u é usado para unsigned inteiro.
O símbolo %c para char.
O símbolo %s para char e cadeias de caracteres.
O símbolo %f para float e o símbolo %lf para double.

Comandos de entrada/saída

```
#include<stdio.h>
int main()
{
    int ano1,ano2;
    printf(“Digite o seu ano de nascimento:");
    scanf(“%d",&ano1);
    printf(“Digite o ano atual:");
    scanf(“%d",&ano2);

    printf(“Voce tem ou tera neste ano %d anos \n",ano2-ano1);
}
```

Comandos de entrada/saída

```
#include<stdio.h>
int main()
{
    float base, altura, area;
    printf`(“Digite a base do retangulo:");
    scanf(“%f",&base);
    printf(“Digite a altura do retangulo:");
    scanf(“%f",&altura);
    area = base* altura;
    printf(“Valor da area e: %f\n",area);
}
```