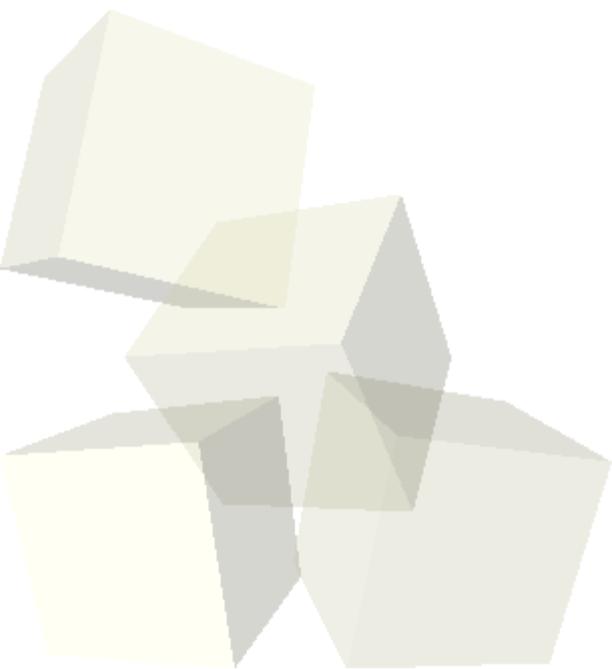




Linguagem C++

aula II - Variáveis e constantes

Prof.: Bruno Gomes





- Representa uma porção da memória que pode ser utilizada pelo programa para armazenar informações
- O nome da variável é utilizado para a sua manipulação no programa
- Toda variável em C++ deve ser “declarada” antes de ser utilizada
- A declaração tem a forma:
`<tipo> var1, var2, ... ,var_n;`
- Exemplos: **int** num;
char ch1, ch2;
double lucro;





Tipos de variáveis

- Variáveis podem ser:
 - ♦ Locais,
 - ♦ parâmetros de funções
 - ♦ ou globais
- Variáveis **locais** são declaradas dentro de um bloco de código
 - ♦ Um bloco é um trecho de código que entre { } (abre-e-fecha chaves)
- **Parâmetros** são variáveis declaradas na assinatura de uma função
 - ♦ São um caso especial de variáveis locais
- Variáveis **globais** são declaradas fora do escopo de qualquer função





Variáveis Locais

- São declaradas dentro de um bloco de comandos
 - ♦ função, estrutura de seleção (if), etc.
 - ♦ O bloco mais comum é a função
- Só existem enquanto o bloco em que elas foram declaradas está sendo executado
 - ♦ Criadas no início da execução do bloco
 - ♦ Após a execução do bloco a variável não fica mais disponível
- Podem ser acessadas por outros blocos
 - ♦ desde que eles sejam internos ao bloco onde a variável foi declarada

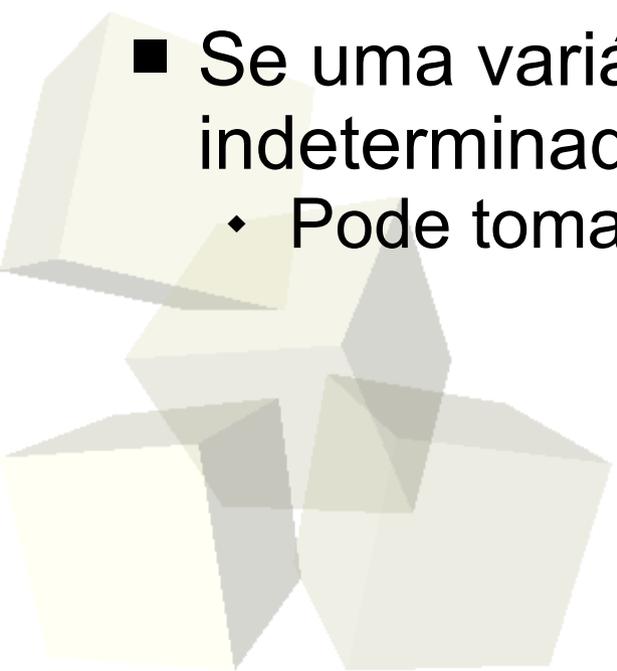


Variáveis Locais: Inicialização

- Podem ser inicializadas:
 - ◆ No momento da declaração
 - ◆ Após a declaração

- Toda vez que o bloco de comandos for executado, a variável será inicializada com o valor dado

- Se uma variável não for inicializada, então seu valor é indeterminado
 - ◆ Pode tomar um valor qualquer que esteja na memória



Exemplos de declarações locais

```
int main() {  
    int i;  
  
    i = 10;  
  
    int j;  
  
    j = 20;  
  
}
```

```
int main(void) {  
    int i;  
  
    i = 10;  
  
    int j = 20;  
  
}
```

```
int main(void) {  
    int i;  
  
    int j;  
  
    i = 10;  
  
    j = 20;  
  
}
```

Exemplos de inicializações locais

```
Int main() {  
    float num;           //num possui valor indefinido  
    char vogal = 'a';   /*vogal é inicializada com 'a' no  
                        momento da declaração*/  
    int a, b, c = 10;    //ERRO: c é inicializada, mas não a e b  
  
    cin >> a; //OK  
    cin >> b; //OK  
}
```



Variáveis Locais: Exemplos

```
int main() {  
    //variável local declarada na função main  
    double saldo;  
}
```

//**x** de “**func1**” não tem relação com **x** de “**func2**”

```
void func1(void) {  
    short int x;  
    x = 10;  
}  
  
void func2(void) {  
    short int x;  
    x = 199;  
}
```

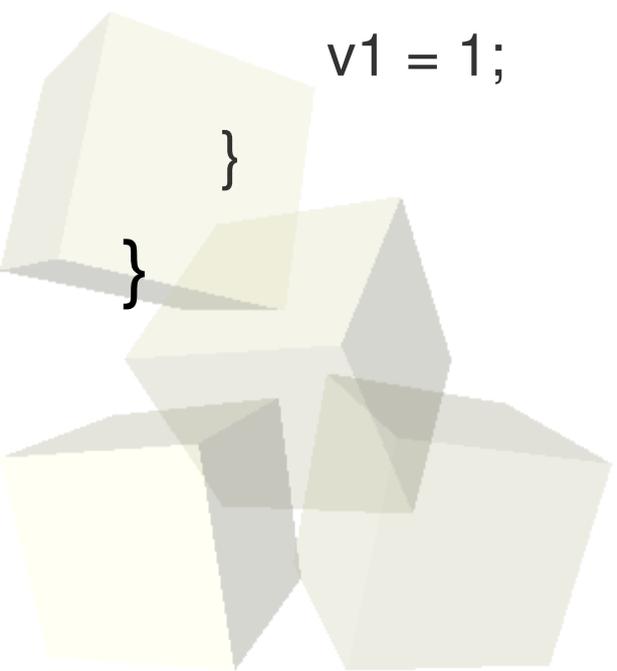




Variáveis Locais: Exemplos

/ v1* pode ser acessada dentro ou fora do *if*, mas *v2* é vista *apenas* dentro do *if* */

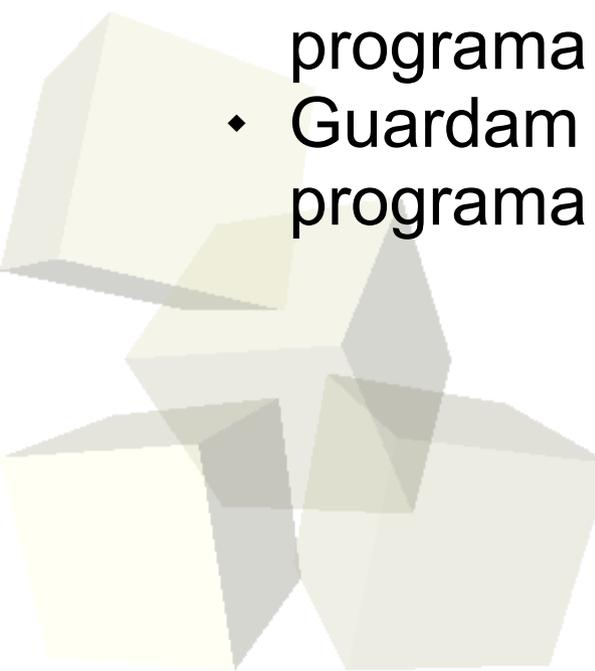
```
void f(void) {  
    int v1;  
  
    if (<alguma_condicao> {  
        char v2[10];  
        v1 = 1;  
    }  
}
```





Variáveis Globais

- Variáveis declaradas fora de qualquer função
 - ◆ Declaradas em qualquer lugar do código, desde que antes do seu primeiro uso
 - ◆ É mais usual a declaração no início do programa
- São reconhecidas pelo programa inteiro
 - ◆ Podem ser acessadas em qualquer bloco dentro do programa
 - ◆ Guardam seus valores durante toda a execução do programa





Variáveis Globais

- Se uma variável *local* é declarada em um bloco com o mesmo nome de uma variável *global*, todas as referências neste bloco referem-se à variável local
- Devem ser evitadas sempre que possível
 - ◆ Ocupam a memória o tempo todo
 - ◆ Dificultam a manutenção do programa
 - ◆ Podem levar a erros difíceis de detectar
- O uso de funções, recebendo e manipulando variáveis locais é mais benéfico





Variáveis Globais: Exemplo

```
...
int contador; ...
int main() {
    contador = 100;
    return 0;
}

void f1() {
    contador = 100; //contador global
}

void f2() {
    int contador;
    contador = 120; //contador local (variável definida dentro da
    função f2)
}
```





O Modificador *const*

- Variáveis ***const*** não podem ser modificadas após a sua inicialização
 - ◆ Exemplo: `const int a = 10;` (cria uma variável inteira, denominada “a” e a inicializada com o valor 10)
- A tentativa de se modificar uma variável *const* gera um erro de compilação
- Principal aplicação:
 - ◆ Quando se quer garantir que uma função não modifique o valor de uma variável passada a ela
- Para a maioria dos casos, é preferível utilizar constantes simbólicas (diretivas)
 - ◆ Maior eficiência





- São valores fixos que o programa não pode alterar
- Em C++ há constantes:
 - ◆ **inteiras, reais, caracteres e strings**
 - ◆
- Constantes **inteiras** são números sem a parte fracionária
- Constantes **reais** são números seguidos de ponto e a parte fracionária
- Uma letra ou código seguido de \ (barra) e entre **aspas simples** representam caracteres especiais
- Uma constante **string** é uma cadeia de caracteres entre aspas duplas





Constantes Inteiras

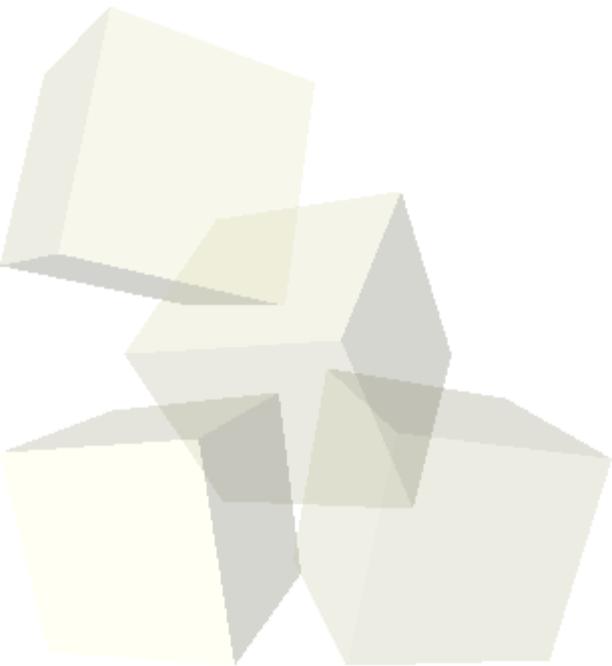
- Números inteiros podem ser escritos na bases decimal (base 10), octal (base 8) e hexadecimal (base 16)
- Constantes decimais: 0, 13, 100, -1781, +98
- Constantes octais são escritas com o número 0 à esquerda do restante do número
 - ♦ Ex.: **00**, **-03**, **-045**, **02633**
- Constantes hexadecimais são identificadas por 0x à esquerda do restante do número
 - ♦ Ex.: **0x0**, **0x56**, **0x64**, **0xA4F2**, **0xde2**





Constantes Reais

- Constantes reais são representadas em base 10 com um ponto decimal e, opcionalmente, um expoente
- Exemplos:
 - ♦ 0.231, 125.65, .93, 1.23e-9, -0.853E+67





Constantes Caracteres

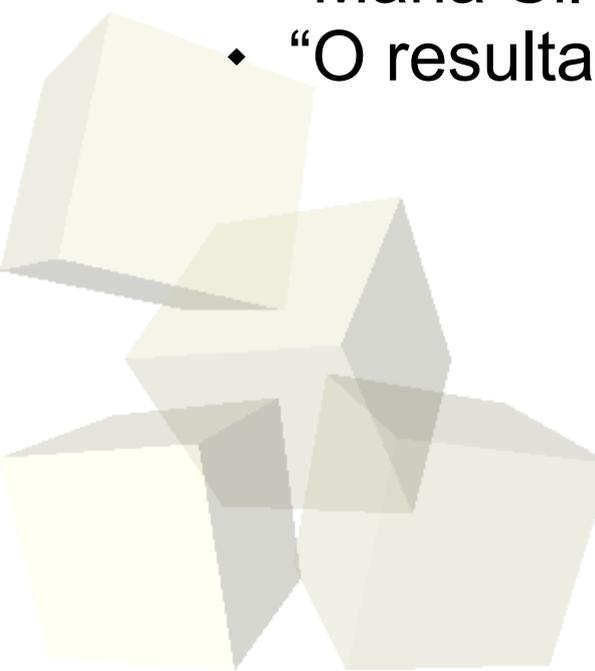
- São letras ou símbolos especiais entre aspas simples ''
- Exemplos: 'a', 'b', 'X', '{', ' ' (espaço em branco), "" (vazio)
- Constantes caracteres são armazenadas internamente como inteiros. 'A' = 65, 'B' = 66 (tabela ASCII)
- Seqüências de escape (caracteres precedidos de \):
 - ♦ '\n' – nova linha
 - ♦ '\t' – tabulação horizontal
 - ♦ '\0' – caracter nulo
 - ♦ '\"' - apóstrofo
 - ♦ '\v' – tabulação vertical
 - ♦ '\\ - barra invertida
 - ♦ '\"' - aspas
 - ♦ '\?' - interrogação





Constantes Strings

- São seqüências de zero ou mais caracteres entre aspas duplas
- Exemplos:
 - ◆ “”
 - ◆ “exemplo\0”
 - ◆ “Maria Silva”
 - ◆ “O resultado da soma é: ” << soma << “\n”





Constantes Simbólicas

- Constantes simbólicas podem ser definidas pelo programador geralmente para substituir “números mágicos” no programa
 - Números mágicos são números que tem um significado, mas estão soltos, sem identificação no programa
 - Para esses propósitos são mais adequadas que variáveis
- Sintaxe da definição:
#define <nome> <valor>
- **#define** é uma diretiva para o pré-processador C++
 - O pré-processador troca cada ocorrência do “nome” da constante no programa pelo seu “valor”
- Usualmente, o nome da constante é colocado em maiúscula





Constantes Simbólicas: Exemplo

```
#define PI 3.14159
int main() {
    ...
    area = PI * raio * raio
    ...
}
```

- O pré-processador C++ substitui, antes da compilação, o nome da constante pelo seu valor:

```
int main() { ...
    area = 3.14159 * raio * raio    ...
}
```