Introdução à Computação II – AULA 07

BCC Noturno - EMA896115B

Prof. Rafael Oliveira
olivrap@gmail.com

Universidade Estadual Paulista
“Júlio de Mesquita Filho”
UNESP

Rio Claro 2014 (Sem 2)

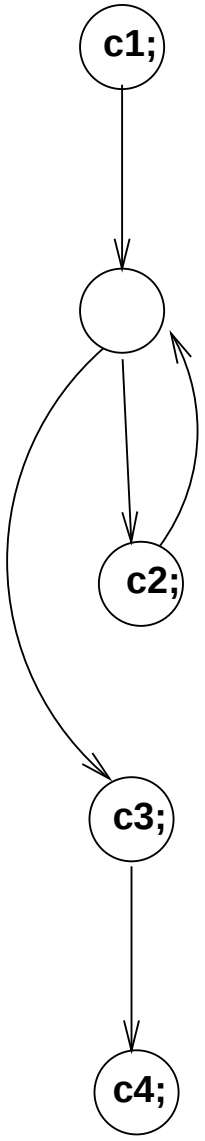
Recordando

- Comando while – repetição
- Strings – usados para guardar cadeias de carcteres, como nome, endereço, etc
- Um string é uma sequência de caracteres legíveis
- Por exemplo char nome[50];
- Vetores de outros tipos também pode ser usados
  - `int v_int[10];`
  - `double x[15];`

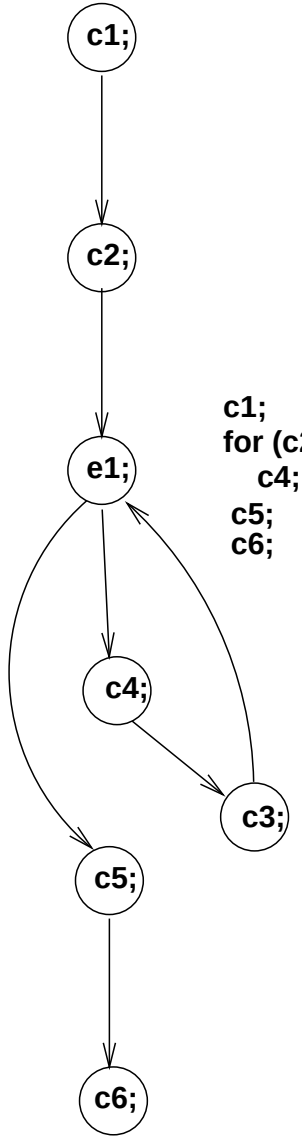
Comando `for`

- O `while` não é o único comando de repetição
- O comando for é parecido mas além da condição incui uma inicialização e um comando de incremento
- ```
  for (i = 0; i < 10; i++)
  {
     printf("%d\n", i);
  }
  ```
- Escreve os números de 0 a 9, um em cada linha
  ```

Representação diagramática




```
c1;
while ( e1 )
    c2;
 c3;
 c4;
```




```
c1;
for (c2; e1; c3 )
    c4;
 c5;
 c6;
```

Equivalência entre os comandos

- Os comandos de repetição `while` e `for` são equivalentes
- O que se faz com um, pode ser frito também com o outro
- Assim, é uma questão de “gosto” usar um ou outro

Equivalência

```
for (i = 0; i < 10; i++)
{
   printf("%d\n", i);
}
```

```
i = 0;
while ( i < 10 )
{
   printf("%d\n", i);
   i++;
}
```

`for` e vetores

- O comando `for` é muito prático quando se trabalha com vetores
- Ele permite que se “percorra” todas as posições do vetor, de maneira simples e fácil de entender
- Uma variável é usada para assumir cada um dos valores do índice do vetor
- ```
  int i;
  double v[10], s;
  s = 0.0;
  for (i = 0; i < 10; i++)
  {
     s = s + v[i];
  }
  ```
  ```

for – exemplo

- Escrever uma função que acha o índice do menor elemento de um vetor

```
int minimo(double v[], int n)
{
int i, min;
min = 0;
for (i = 0; i < n; i++)
{
   if ( v[i] < v[min] )
   {
      min = i;
   }
   return min;
}
```

`for` – exemplo (2)

- Função que remove de um vetor todas as ocorrências de um determinado valor
- Exemplo: remover o valor 5 do vetor $\langle 1, 2, 5, 6, 7, 5, 3 \rangle$ produz $\langle 1, 2, 6, 7, 3 \rangle$
- Retorna o tamanho do vetor modificado. No exemplo: 5

for – exemplo (2)

```
int remover(int v[], int r, int n)
{
int i, j;
   for (i = 0; i < n; i++)
   {
       if ( v[i] == r )
       {
          for (j = i; j < n - 1; j++)
          {
              v[j] = v[j+1];
          }
          n--;
       }
   }
   return n;
}
```

Exercício

- Transforme o programa anterior, usando apenas o comando `while`

Exercício

- Transforme o programa anterior, usando apenas o comando `while`

```
int remover(int v[], int r, int n)
{
int i, j;
   i = 0;
   while ( i < n)
   {
       if ( v[i] == r )
       {
          j = i;
          while ( j < n - 1)
          {
              v[j] = v[j+1];
              j++;
          }
          n--;
       }
       i++;
   }
   return n;
}
```

for – exemplo(3)

- Escrever uma função que verifica se um dado número inteiro é primo. Deve retornar 1 se for primo e 0 se não for

for – exemplo(3)

- Escrever uma função que verifica se um dado número inteiro é primo. Deve retornar 1 se for primo e 0 se não for

```
int primo(int n)
{
int i, eh_primo;
   eh_primo = 1;
   if ( n < 2 ) return 0;
   for (i = 2; i < n && eh_primo; i++)
   {
       if ( n % i == 0 )
          eh_primo = 0;
   }
   return eh_primo;
}
```

Matrizes

- Quando um array tem duas (ou mais) dimensões nós o chamamos de **matriz**. Para referenciar um elemento de uma matriz é preciso fornecer o índice da linha e o índice da coluna do elemento

$$\begin{pmatrix} a_{0.0} & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} & a_{0,5} & a_{0,6} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} & a_{1,6} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} & a_{2,6} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & a_{3,6} \end{pmatrix}$$

Matrizes

- Quando um array tem duas (ou mais) dimensões nós o chamamos de **matriz**. Para referenciar um elemento de uma matriz é preciso fornecer o índice da linha e o índice da coluna do elemento

0,0	0,1	0,2	0,3	0,4	0,5	0,6
1,0	1,1	1,2	1,3	1,4	1,5	1,6
2,0	2,1	2,2	2,3	2,4	2,5	2,6
3,0	3,1	3,2	3,3	3,4	3,5	3,6

Matrix × vetor

- double x[7];
-

0	1	2	3	4	5	6

- double y[4][7];
-

0,0	0,1	0,2	0,3	0,4	0,5	0,6
1,0	1,1	1,2	1,3	1,4	1,5	1,6
2,0	2,1	2,2	2,3	2,4	2,5	2,6
3,0	3,1	3,2	3,3	3,4	3,5	3,6

- `s = y[1][3] + y[3][0];`

Matriz – exemplo

- Escrever uma função que zera uma matriz

```
void zera(double m[][], int n, int k)
{
int lin, col;

    for (lin = 0; lin < n; lin++)
    {
        for (col = 0; col < k; col++)
        {
            m[lin][col] = 0.0;
        }
    }
}
```

Matriz – exemplo

- Escrever uma função que zera uma matriz.
- É preciso saber o tamanho de cada linha. Apenas o número de linhas pode ser deixado sem valor

```
void zera(double m[][10], int n)
{
int lin, col;

    for (lin = 0; lin < n; lin++)
    {
        for (col = 0; col < 10; col++)
        {
            m[lin][col] = 0.0;
        }
    }
```

Alternativa

- Uma alternativa é usar um tamanho máximo de linha, mesmo que os últimos elementos não sejam usados

```
void zera2(double m[][100], int l, int c)
{
int lin, col;
    for (lin = 0; lin < l; lin++)
    {
        for (col = 0; col < c; col++)
        {
            m[lin][col] = 0.0;
        }
    }
}
int main()
{
double matriz1[5][10], matriz2[100][100];
   zera(matriz1, 5);
   zera2(matriz2, 5, 10);
}
```

Matriz – exemplo (2)

- Escrever uma função que leia uma matriz de tamanho $n \times m$

```
#include <stdio.h>
#define MAX 100
double matriz1[MAX][MAX];
void le_matriz(double m[][MAX], int l, int c)
{
int lin, col;
    for (lin = 0; lin < l; lin++)
    {
        for (col = 0; col < c; col++)
        {
            printf("Elemento [%3d,%3d] ==> ", lin, col);
            scanf("%lf", &m[lin][col]);
        }
    }
}
```

Matriz – exemplo (2)

- Escrever uma função que leia uma matriz de tamanho $n \times m$

```
int main()
{
int n, m;
   printf("Numero de linhas: ");
   scanf("%d", &n);
   printf("Numero de colunas: ");
   scanf("%d", &m);
   le_matriz(matriz1, n, m);
}
```

Matriz – exemplo(3)

```
void print_matriz(double m[][MAX], int l, int c)
{
int lin, col;
    for (lin = 0; lin < l; lin++)
    {
        for (col = 0; col < c; col++)
        {
            printf("%5.2lf ", m[lin][col]);
        }
        printf("\n");
    }
}
```

Matriz – exemplo(3)

```
void soma(double m1[][MAX], double m2[][MAX],
          double result[][MAX], int l, int c)
{
int lin, col;

    for (lin = 0; lin < l; lin++)
    {
        for (col = 0; col < c; col++)
        {
            result[lin][col] = m1[lin][col] +
                               m2[lin][col];
        }
    }
}
```

Matriz – exemplo(3)

```
double matriz1[MAX][MAX],
       matriz2[MAX][MAX],
       matriz3[MAX][MAX];
int main()
{
int n, m;
   printf("Numero de linhas: ");
   scanf("%d", &n);
   printf("Numero de colunas: ");
   scanf("%d", &m);
   le_matriz(matriz1, n, m);
   le_matriz(matriz2, n, m);
   soma(matriz1, matriz2, matriz3, n, m);
   print_matriz(matriz3, n, m);
}
```