Universidade de São Paulo - São Carlos,SP
Instituto de Ciências Matemáticas e de Computação

USP - ICMC - SSC
SSC 0301 - 2o. Semestre 2013

Disciplina de
Introdução à Computação
para Engenharia Ambiental

Prof. Dr. Fernando Santos Osório
LRM - Laboratório de Robótica Móvel do ICMC / CROB-SC
Email: fosorio@icmc. usp. br ou fosorio@gmail. com
Página Pessoal: http://www.icmc.usp.br/~fosorio/

Material on-line:
Wiki ICMC - http://wiki.icmc.usp.br/index.php
Wiki SSC0301 - http://wiki.icmc.usp.br/index.php/SSC-301-2013(fosorio)

Aula 06

Linguagem de Programação "C"

Agenda:

- Comandos de Repetição: FOR e WHILE (relembrando!)
- Vetores: Agrupando variáveis do mesmo tipo.

  Vetor = variáveis compostas (em seqüência) homogêneas
- Vetores de: int, float, double, char
- Declaração e uso de Vetores

  > Exercícios

Informações Complementares a Atualizadas:
Consulte REGULARMENTE o material disponível na WIKI
http://wiki.icmc.usp.br/index.php/SSC-301-2013(fosorio)

Fluxo de Execução: Laço de Repetição

Comando de Laço: FOR

- CONTAR e EXECUTAR 10 vezes algo (contar até 10)

```
/* Calcula a média simples de 2 notas para uma turma de 10 alunos */
main()
{
   float Nota1, Nota2, Media;
   int   aluno;

   for  (aluno = 1; aluno <= 10; aluno ++)
   {
        printf("Digite a Nota1: ");
        scanf("%f",&Nota1);
        printf("Digite a Nota2: ");
        scanf("%f",&Nota2);
        Media = (Nota1 + Nota2) / 2.0;
        printf ("Aluno %d - Media = %.2f\n", aluno, Media);
   }
}
```

Fluxo de Execução: Laço de Repetição

Comando de Laço: WHILE

- CONTAR e EXECUTAR 10 vezes algo (contar até 10)

```
/* Calcula a média simples de 2 notas para uma turma de 10 alunos */
main()
{
   float Nota1, Nota2, Media;
   int   aluno;
   aluno=1;
   while (aluno <= 10)
   {
        printf("Digite a Nota1: ");
        scanf("%f",&Nota1);
        printf("Digite a Nota2: ");
        scanf("%f",&Nota2);
        Media = (Nota1 + Nota2) / 2.0;
        printf ("Aluno %d - Media = %.2f\n", aluno, Media);
        aluno ++;
   }
}
```

Fluxo de Execução: Laço de Repetição

Comando de Laço: WHILE

- CONTAR e EXECUTAR 10 vezes algo (contar até 10)

```
/* Calcula a média simples de 2 notas para uma turma de 10 alunos */

main()
{
   float Nota1, Nota2, Media;
   int   aluno;

   aluno=1;
   while (aluno <= 10)
   {
        printf("Digite a Nota1: ");
        scanf("%f",&Nota1);
        printf("Digite a Nota2: ");
        scanf("%f",&Nota2);
        Media = (Nota1 + Nota2) / 2.0;
        printf ("Aluno %d - Media = %.2f\n", aluno, Media);
        aluno ++;
   }
}
```

E se fosse necessário "guardar" as notas de todos alunos?

E se cada aluno tivesse 10 notas? N1, N2, N3, ... N10?

Linguagem "C": VETORES

VETORES: Agrupando Dados Iguais em Seqüência

E se eu precisasse declarar 10 Notas?
Teria que criar 10 variáveis ?!?

```
double   Nota1,  Nota2,  Nota3,  Nota4,  Nota5,
         Nota6,  Nota7,  Nota8,  Nota9,  Nota10;   (UFA!!!)
```

E se eu precisasse ler estas 10 Notas?
Teria que criar um conjunto de 10 printf/scanf ?!?

```
printf ("Entre com a nota 1: ");
scanf  ("%lf",&Nota1);
printf ("Entre com a nota 2: ");
scanf  ("%lf",&Nota2);
...
printf ("Entre com a nota 10: ");
scanf  ("%lf",&Nota10);
```

Linguagem "C": VETORES

VETORES: Agrupando Dados Iguais em Seqüência

E se eu precisasse declarar 10 Notas?
Teria que criar 10 variáveis ?!?

```
double   Nota1,  Nota2,  Nota3,  Nota4,  Nota5,
         Nota6,  Nota7,  Nota8,  Nota9,  Nota10;   (UFA!!!)
```

E se eu precisasse ler estas 10 Notas?
Teria que criar um conjunto de 10 printf/scanf ?!?

```
printf ("Entre com a nota 1: ");
scanf  ("%lf",&Nota1);
printf ("Entre com a nota 2: ");
scanf  ("%lf",&Nota2);
...
printf ("Entre com a nota 10: ");
scanf  ("%lf",&Nota10);
```

E se eu fosse ler
a cotação do dólar
nos últimos 30, 60, 90 dias?!?

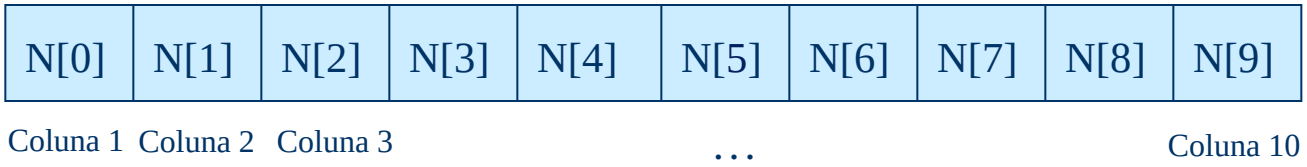
Linguagem "C": VETORES

VETORES: Agrupando Dados Iguais em Seqüência

E se eu precisasse declarar 10 Notas?
Teria que criar 10 variáveis ?!?

Nota1, Nota2, Nota3, Nota4, Nota5,
Nota6, Nota7, Nota8, Nota9, Nota10 (UFA!!!)

Nestes casos podemos usar um VETOR ou uma seqüência de variáveis formando uma lista onde eu indico o seu nome e o seu índice (coluna) que eu desejo acessar



Vetor

Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

Exemplos Típicos:

VETOR DE CARACTERES = *STRING*

```
char   Texto[10];
```

Texto[0] até Texto[9] <= São 10 posições de 1 char, lado a lado

---

VETOR DE INTEIROS = *TABELA*

```
int    Tabela[10];
```

Tabela[0] até Tabela[9] <= São 10 posições de 1 int, lado a lado

---

VETOR DE DOUBLES = *DADOS*

```
double  Dados[10];
```

Dados[0] até Dados[9] <= São 10 posições de 1 double, lado a lado

Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

Vetores Numéricos:

```
double N[10];      /* Notas de até 10 alunos */

N[0] = 10.0;
N[1] = 5.0;
N[2] = 7.77;

Qtde_Notas = 3;      /* Última = Qtde_Notas – 1 */

N[Qtde_Notas] = 9.0;          /* Nota índice 3  */
Qtde_Notas++;
N[Qtde_Notas++] = 8.0;      /* Nota índice 4  */
Qtde_Notas++;
```

N[0]	N[1]	N[2]	N[3]	N[4]	N[5]	N[6]	N[7]	N[8]	N[9]
10.0	5.0	7.77	9.0	8.0					

Qtde_Notas

Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

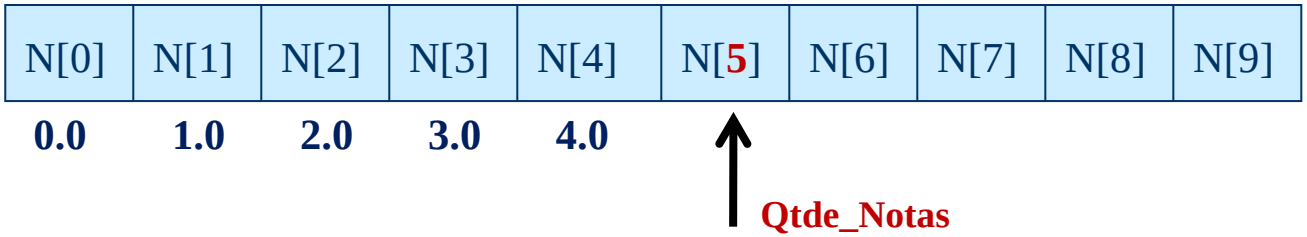
VETORES

Vetores Numéricos:

```
double N[10];      /* Notas de até 10 alunos */

int i;
int Qtde_notas;

Qtde_Notas = 5;
for  (i = 0;  i < Qtde_Notas; i++ )  {
        printf ("Entre com a nota %d: ", i );
        scanf ("%lf", &N[i] );
}
```



Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

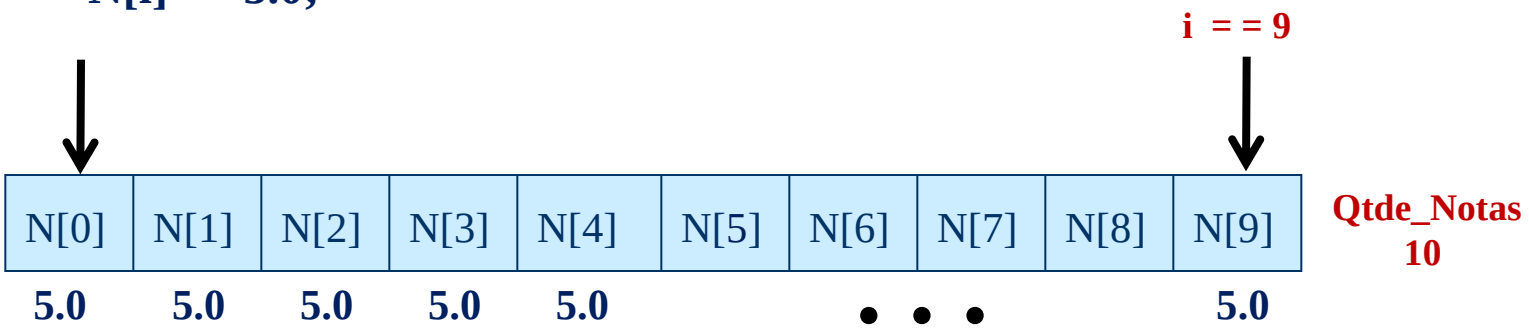
VETORES

Vetores Numéricos:

```
double N[10];      /* Notas de até 10 alunos */

int i;
int Qtde_notas;

Qtde_Notas = 10;
for  (i = 0;  i < Qtde_Notas; i++ )
      N[i]  =  5.0;
```



Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

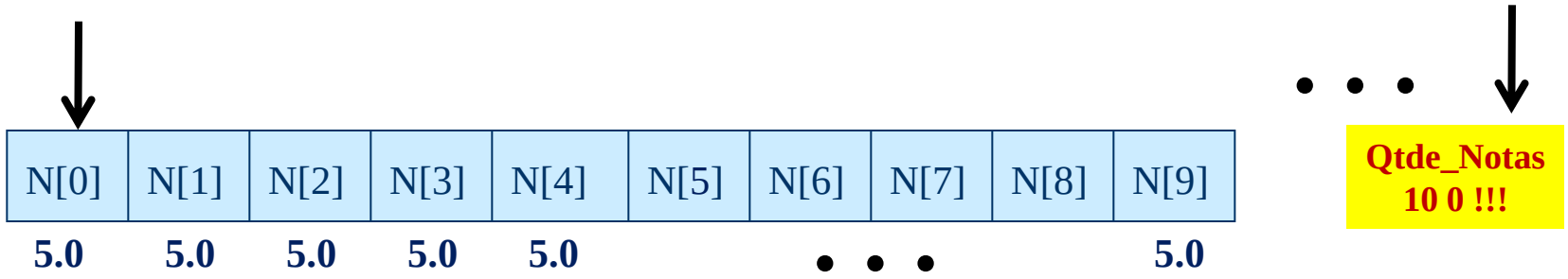
Vetores Numéricos:

```
double N[10];      /* Notas de até 10 alunos */

int i;
int Qtde_notas;

Qtde_Notas = 100;
for  (i = 0;  i < Qtde_Notas; i++ )
      N[i]  =  5.0;
```

Atenção:
A linguagem "C" não controla se usarmos um índice INVALIDO! (causa um "estouro de memória")



Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

double N[10];

int i;
int Qtde_notas;

N[0] = 0.0001;
N[5] = 0.001;
N[9] = 0.01;

N[i] = 5.0;
N[Qtde_notas]=0.0;

Lembretes sobre os "INDICES" de um Vetor:

- O índice pode ser uma variável...
- O valor armazenado é do tipo declarado (ex. double) mas o índice tem que ser INTEIRO (valor escalar, ou seja, contável/enumerável).
- O índice do vetor NÃO pode ser um float ou double!
- Os índices SEMPRE começam em ZERO [0.. X].
- A linguagem "C" não controla índices fora da faixa de valores declarada.

~~**N[10] = 1.0;**~~ **ERRADO! Não existe o N[10] em um vetor como o declarado acima [0..9]**

N[0]	N[1]	N[2]	N[3]	N[4]	N[5]	N[6]	N[7]	N[8]	N[9]

Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

Vetores de Caracteres:

```
char T[10];      /* String de até 9 caracteres  + Null */

T[0] = 'O';
T[1] = 'L';
T[2] = 'A';
T[3] = '\0';     /* Caracter  NULL ou \0  */
```

Reservado para o '\0'


```
strcpy(T,"OLA");
/* Inclui o Null automaticamente */
/* ao final da String            */
```

Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

Vetores de Caracteres:

char Texto[10]; /* String de até 9 caracteres + Null */

STRINGS são vetores de Caracteres!

Isso explica coisas como...

```
scanf ("%d", &Valor);      // Para ler um inteiro (ou double) tem o "&"
scanf ("%s", Texto);       // Exceção: não precisa do "&" em strings!
```

veremos que o comando acima equivale a...

```
scanf( "%s", &(Texto[0] ) );   // Endereço do primeiro caracter da string
```

T[0]	T[1]	T[2]	T[3]	T[4]	T[5]	T[6]	T[7]	T[8]	T[9]
'H'	'E'	'L'	'L'	'O'	'\0'				

Null

Final da String

Linguagem "C": VETORES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES

Vetores de Caracteres:

char Texto[10]; /* String de até 9 caracteres + Null */

Strings **precisam** ser manipuladas através de rotinas especiais:
#include <string.h>
strcpy, strlen, strcmp... sprintf, sscanf, ...

T[0]	T[1]	T[2]	T[3]	T[4]	T[5]	T[6]	T[7]	T[8]	T[9]
'H'	**'E'**	**'L'**	**'L'**	**'O'**	**'\0'**				
					Null				
					Final da String				

Exercícios – Laço de Repetição + Vetores

EXERCÍCIOS: USANDO FOR e VETORES

- Faça um programa que leia um conjunto de 10 notas, armazenando em um vetor. Uma vez lidos os valores, exibir na ordem inversa em que foram lidos os dados, ou seja, o último dado a ser exibido na tela deve ser o primeiro que foi lido. Exemplo:
  > Digite 10 Notas: 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0
  > Notas Lidas: 10.0 9.0 8.0 7.0 6.0 5.0 4.0 3.0 2.0 1.0

- Altere o programa anterior de modo a validar as notas fornecidas. Uma nota deve ser um valor entre 0 e 10, sendo que o programa deve repetir a leitura das notas até que o usuário digite uma nota válida.

- Baseado no programa anterior, uma vez concluída a leitura das notas, exibir na tela o maior, o menor valor e a média dentre os valores que foram lidos, juntamente com o seu índice. Exemplo:
  > Digite 10 Notas: 6.0 6 .1 6.2 6.3 9.2 6.5 6.7 6.8 6.9 3.4
  > Maior Nota: Nota[4] = 9.2
  > Menor Nota: Nota[9] = 3.4
  > Média das Notas: 6.41

VETORES e MATRIZES

- **O QUE MAIS PODEMOS FAZER COM VETORES...**

- **Não percam.... Cenas do Próximo Capítulo!**

VETORES e MATRIZES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

VETORES - Vetores com UMA dimensão
Sequência de Dados

N[0]	N[1]	N[2]	N[3]	N[4]	N[5]	N[6]	N[7]	N[8]	N[9]

Coluna 1 Coluna 2 Coluna 3 ... Coluna 10

Vetor

MATRIZES - Vetores com mais de uma dimensão
TABELAS de Dados

Matriz

N[0]	N[1]	N[2]	N[3]	N[4]	N[5]	N[6]	N[7]	N[8]	N[9]	Linha 1
N[0]	N[1]	N[2]	N[3]	N[4]	N[5]	N[6]	N[7]	N[8]	N[9]	Linha 2
N[0]	N[1]	N[2]	N[3]	N[4]	N[5]	N[6]	N[7]	N[8]	N[9]	Linha 3

Coluna 1 Coluna 2 Coluna 3 ... Coluna 10

VETORES e MATRIZES

Tipos de Dados Compostos: Estruturas HOMOGÊNEAS

MATRIZES - Vetores com mais de uma dimensão

Vetores numéricos bi-dimensionais:

3 x 10

```
int  Matriz [3][10];

Matriz[0][0] = 1;
...
Matriz [2][9] = 30;
```

M[0][0]	M[0][1]	M[0][2]	M[0][3]	M[0][4]	M[0][5]	M[0][6]	M[0][7]	M[0][8]	M[0][9]
M[1][0]	M[1][1]	M[1][2]	M[1][3]	M[1][4]	M[1][5]	M[1][6]	M[1][7]	M[1][8]	M[1][9]
M[2][0]	M[2][1]	M[2][2]	M[2][3]	M[2][4]	M[2][5]	M[2][6]	M[2][7]	M[2][8]	M[2][9]

- Inicialização de vetores:

```
int num [5] =  { 1, 2, 3, 4, 5 };

char vogais[5]  = { 'a', 'e', 'i', 'o', 'u' };

double matriz [3][2]  = {  { 0,0 }, { 0,1 },
                           { 1,0 }, { 1,1 },
                           { 2,0 }, { 2,1 }  };
```

Matriz do Jogo da Velha

```
char   Tabuleiro [3][3];
```

'O'	'X'	'X'
' '	'O'	' '
' '	' '	'O'

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/
Página pessoal: http://www.icmc.usp.br/~fosorio/
Página do Grupo de Pesquisa: http://www.lrm.icmc.usp.br/
E-mail: fosorio [at] icmc. usp. br ou fosorio [at] gmail. com

Disciplina de Introdução a Computação – Eng. Ambiental
WIKI - http://wiki.icmc.usp.br/index.php/SSC-301-2013(fosorio)
> Programa, Material de Aulas, Critérios de Avaliação,
> Trabalhos Práticos, Datas das Provas, Notas