Departamento de Sistemas de Computação
Universidade de São Paulo
SSC 0601 – Laboratório de Introdução à Ciência de Computação I

# Aula 4
# Estrutura de Repetição e Vetores (+ Strings)

**Responsável**

Prof. Armando Toda (armando.toda@usp.br)

# Revisão

# Estrutura condicional

# Revisão: Estrutura condicional

- As condições de decisão devem ser compostas por comparações entre 2 elementos
  - SE a < b ENTAO ....
  - SE y > 2 ENTAO ....
- As condições de decisão podem ser compostas por múltiplas comparações
  - SE a < b  OU  b < c ENTAO
  - SE x < 2  E  x  > 0 ENTAO  ( 0 < x < 2 --- não vale)
- **Crie casos de teste para verificar se sua solução é correta**
- **Quanto melhor você entender o problema mais bonita será a sua solução**

Dado três inteiros crie um algoritmo para retornar o menor deles

# Resposta

```
LEIA n1, n2, n3
SE (n1 <= n2) ENTAO
    SE (n1 <= n3) ENTAO
        IMPRIME n1
    SENAO
        IMPRIME n3
    FIMSE
SENAO
    SE (n2 <= n3) ENTAO
        IMPRIME n2
    SENAO
        IMPRIME n3
    FIMSE
FIMSE
```

## E como fazer ??????

# Escrevemos 10 linhas de código para encontrar o menor valor dentre 3 números

## E como fazer ??????

Mas e se você quiser achar o menor valor dentre 10, 100 ou 1000 números?

O que fazer?

# Como decompor o problema?

## E como fazer ??????

# 1) Como armazenar e organizar os números?

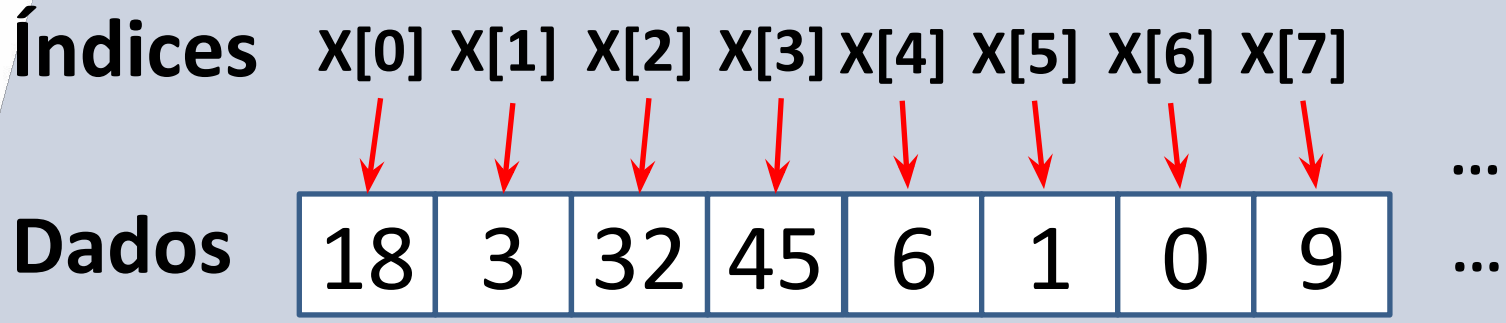
ICMC USP SÃO CARLOS

## ÍNDICE


| | | |
|---|---|---|
| **1** | **INTRODUÇÃO** | 7 |
| | 1.1 A TECNOLOGIA DA INFORMAÇÃO NA COMUNIDADE ACADÊMICO-CIENTÍFICA | 13 |
| | 1.2 DEFININDO A INTERNET | 18 |
| | 1.2.1 Números da Internet no Brasil | 24 |
| | 1.3 A COMUNICAÇÃO DO CONHECIMENTO CIENTÍFICO | 26 |
| | 1.4 A LITERATURA CIENTÍFICA SOBRE A INTERNET | 36 |
| **2** | **OBJETIVOS** | 41 |
| **3** | **MÉTODOS** | 42 |
| | 3.1 UNIVERSO DE ESTUDO | 42 |
| | 3.2 COLETA DE DADOS | 47 |
| | 3.2.1 Instrumento de Coleta de Dados | 47 |
| | 3.2.2 Procedimento de Envio dos Questionários | 50 |
| | 3.3 ANÁLISE ESTATÍSTICA | 52 |
| | 3.4 QUESTÕES ÉTICAS | 52 |
| **4** | **RESULTADOS** | 53 |
| | 4.1 CARACTERIZAÇÃO DA POPULAÇÃO ESTUDADA | 59 |
| | 4.2 USO DA INTERNET PELA COMUNIDADE ESTUDADA | 60 |
| | 4.2.1 Uso dos Recursos da Internet | 61 |


Índices permitem encontrar um dado em um conjunto de dados

# Vetores (Arrays)

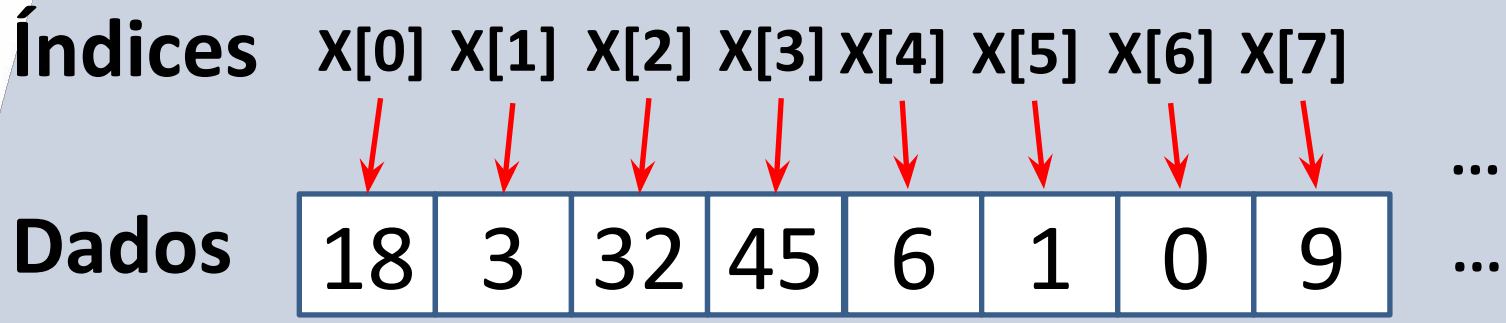



## Vetores

são estruturas de dados que armazenam diversos valores de um mesmo tipo

- Se acessar o índice x[2] o dado retornado será 32

# Vetores (Arrays)




## Declaração

1) <tipo> x[10]

Vetor com tamanho definido 10. Obs: valor acessíveis do vetor são de 0 a 9

2) <tipo> x[]

Vetor de tamanho qualquer

# 2) Como achar o menor valor dentro de um conjunto de dados ?

# Como saber qual das cartas do conjunto é a menor?

# Faça um algoritmo para comparar duas cartas

1. Pegue duas cartas
2. Compare as cartas
3. Fique com a carta menor
4. Descarte a carta maior

```
int menorCarta, c1, c2;
scanf("%d %d", &c1, &c2);
if (c1 < c2) {
   menorCarta = c1;
} else {
   menorCarta = c2;
}
```

# Faça um algoritmo para compara um conjunto de cartas

1. Pegue duas cartas
2. Compare as cartas
3. Fique com a carta menor
4. Descarte a carta maior
5. **Pegue outra carta**
6. **Repita o passo 2 até acabar as cartas**

```
int menorCarta, c1, c2

if (c1 < c2) {
    menorCarta ← c1
} else {
    menorCarta ← c2
}
```

```
int menorCarta, c1, c2;
int vCartas[10];  //suponha que já existe
                  //   um vetor de cartas
if (c1 < c2){
    menorCarta = c1
} else {
    menorCarta = c2;
}
```

```
int menorCarta;
int vCartas[10];  //suponha que já existe
                  //   um vetor de cartas

scanf("%d", &vCartas[0]);
(...)
if ((vCartas[0] < vCartas[1]) &&
(vCartas[0] < vCartas[2]) && (...) )
```

```
int vCartas[10], menor_carta=13;
for (int i = 0; i < 10; i++){
    scanf("%d",&vCartas[i]);
    if(menor_carta <= vCartas[i]){
        menor_carta = vCartas[i];
    }
}
printf("Menor carta: %d", menor_carta);
```

# Isso é uma Estrutura de Repetição

# Comandos

- Declaração de variáveis:
  - int, float, char
- Leitura/Escrita
  - Leitura de dados: scanf()
  - Escrita de dados: printf()
- Estrutura Condicional
  - Simples: if()-else
  - Composta: if()-else if()-else
- **Estrutura de Repetição**
  - **for()**
  - **while()**
  - **do…while()**

# Estrutura de Repetição

- Uma estrutura de repetição é utilizada quando um comando ou um bloco de comandos deve ser repetido.
- A quantidade de repetições pode ser fixa ou pode depender de uma determinada condição.
- O teste da condição pode ocorrer no início ou no final da estrutura de repetição.

- Três tipos de estruturas serão consideradas na elaboração de Algoritmos:
  - **Estrutura PARA**
  - **Estrutura ENQUANTO**
  - **Estrutura REPITA**

# Estrutura de Repetição: FOR

```
for(i = valorInicial; parada ; incremento/decremento) {
    Instrução_1
    Instrução_2
    ....
    Instrução_n
}
```

# Estrutura de Repetição: FOR

- Normalmente utilizada quando é conhecido o número de repetições.
- A variável i é utilizada como controle, variando do valor_inicial até valor_final.
- Possui um incremento automático

# Estrutura de Repetição: FOR

- Exemplos:

```
for(int i = 1; i <= 10; i++ ){
    printf("%d ", i);
}
```

**Saida: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10**

```
for(int i = 10; i >=5; i--){
    printf("%d ", i);
}
```

**Saida: 10, 9, 8, 7, 6, 5**

# Exercício

- Faça um algoritmo que exiba na tela uma contagem de 0 até 30, exibindo apenas os múltiplos de 3.
- Faça um algoritmo que leia um conjunto de 10 valores, 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.

# Representação de Algoritmos

- Declaração de variáveis:
  - DECLARE
- Leitura/Escrita
  - Leitura de dados: LEIA
  - Escrita de dados: IMPRIMA
- Estrutura Condicional
  - Simples: SE-ENTAO
  - Composta: SE-ENTAO-SENAO
- **Estrutura de Repetição**
  - **PARA**
  - **ENQUANTO**
  - **REPITA**

# Estrutura de Repetição: WHILE

```
while(condiçãoVerdadeira) {
    Instrução_1
    Instrução_2
    ....
    Instrução_n
}
```

# Estrutura de Repetição: while

Quantas vezes o código abaixo vai ser repetido?

```
x = 1;
y = 5;
while (x < y){
    x = x+2;
    y = y+1;
}
```

# Exercício

- Chico tem 1,50 metro e cresce 2 centímetros por ano, enquanto Zé tem 1,10 metro e cresce 3 centímetros por ano. Construa um algoritmo que calcule e imprima quantos anos serão necessários para que Zé seja maior que Chico.

# Exercício

- Ler um número inteiro n. Escrever a soma de todos os números pares de 2 até n (Dica: Utilize o operador de resto %).

# Estrutura de Repetição: while

- Normalmente utilizada quando não se sabe exatamente o número de repetições.
- Também pode ser utilizada quando o número de repetições é conhecido.
- A repetição é executada enquanto a condição for **verdadeira**.

# Estrutura de Repetição: do-while

```
do{
    Instrução_1
    Instrução_2
    ...
    Instrução_n
}while (condição);
```

```
i = 1;
media = 0;
do{
   printf("Nota prova: ");
   scanf("%f",&nota);

   media = (nota + media)/i;
   printf("Media do aluno = %f\n",media);
   i++;
   printf("Digite 1 para adicionar nota ou 0 para sair\n");
   scanf("%d", &resp);
 } while (resp == 1);
```

# Estrutura de Repetição: do-while

- Normalmente utilizada quando não se sabe exatamente o número de repetições.
- Também pode ser utilizada quando o número de repetições é conhecido.
- A repetição é executada ATÉ que a condição se torne verdadeira.
- A diferença entre a estrutura REPITA e ENQUANTO é que as instruções em REPITA serão executadas ao menos uma vez.

# (+ Strings)

# Strings

- Até o momento não lidamos com palavras inteiras, apenas com caracteres individuais
- Uma string é um vetor de caracteres

**char nome[10];** //um vetor de caracteres chamado nome, contendo 10 posiçoes

# Strings

```
char nome[10] = "Armando";
char nome[10] = {'A','r','m','a','n','d','o'};

scanf("%s",nome); //lê até um espaço

gets(nome); //lê até um enter
```

# Strings

Acessando os caracteres da string

```
for(int i = 0; i<10; i++){
    printf("%c ", nome[i]);
}

printf("%c", nome[10]);
```

# Strings

Comparação de strings

```
char nome1[10] = "Armando";
char nome2[10] = "Toda";

if(nome1 == nome2){
...
}
```

# Strings

Comparação de strings

```
#include <string.h>
(...)
char nome1[10] = "Armando";
char nome2[10] = "Toda";
ret = strcmp(nome1,nome2);
if(ret == 0){
printf("São iguais");
}else printf("são diferentes");
```

# Exercícios

# Exercício

Seja a seguinte série:

1, 4, 9, 16, 25, 36, ...

Escreva um algoritmo que gere esta série até o N-ésimo termo. Este N-ésimo termo é digitado pelo usuário.

# Exercício 2

Num frigorífico existem 90 bois. Cada boi traz preso em seu pescoço um cartão contendo seu número de identificação (1 até 90) e seu peso. Faça um algoritmo que escreva o número e o peso do boi mais gordo e do boi mais magro (supondo os bois não tem pesos iguais).

**[opcional]** E se o número de identificação for qualquer número? Como você modificaria o algoritmo acima?