

---

# Laboratório de Introdução à Ciência da Computação I

## **Aula 3 – Estrutura de Repetição (continuação) e Vetores**

Professor:  
Jó Ueyama

# Sumário

---

- Comandos de *loop* (continuação)
  - for (visto na aula anterior)
  - while
  - do-while
- Comando de controle de laço
  - break
  - continue
  - goto
- Vetores e strings
- Exercícios

# Comando while

---

```
while (expressão de teste)
    instrução;
```

```
while (expressão de teste)
{
    instrução_1;
    instrução_2;
    ...
    instrução_n;
}
```

# Comando while

---

- O “while” pode substituir o “for” do seguinte modo:

Inicialização da variável de teste

```
while(teste)
```

```
{
```

```
Incremento da variável de teste;
```

```
....
```

```
}
```

# Comando while

---

- Exemplo 1

```
int main()
{
    int conta=0;
    int total=0;
    while(conta<10)
    {
        total+=conta;
        printf("conta=%d, total=%d\n", conta, total);
        conta++;
    }
    system("pause");
}
```

# Comando while

---

- Exemplo 2

```
int main()
{
    int conta=0;
    printf("Digite uma frase:\n");
    //13 é o valor do caractere "enter" ou "cr"(carriage return)
    while(getche()!=13)
    {
        conta++;
    }
    printf("A frase possui %d caracteres",conta);
    system("pause");
}
```

Veja a tabela ASCII: <http://www.asciitable.com/>

# Comando while aninhado

---

```
while (expressão de teste1)
    while(expressão de teste2)
        instrução;
```

- Pode existir n comandos while aninhados
- Incremente o programa da calculadora da aula 2 para permitir que o usuário realize operações até que a tecla “Esc” for pressionada para sair

# Comando do-while

---

```
do
{
    instrução;
}while(expressão de teste);
```

- Permite executar o bloco mesmo se o teste for falso no início
- Pouco utilizado



# Comando break

---

- Pode ser utilizado no corpo de qualquer estrutura de laço;
- Causa a saída imediata do laço desviando o programa para a próxima instrução após o laço atual
- Se estiver em laços aninhados, o break afetará somente o laço que o contém e seus laços internos

# Comando continue

---

- Força a próxima iteração do laço, não executando o código que vem a seguir
- No caso do “while” e “do-while” a execução é desviada para o teste condicional e depois segue para o corpo do laço
- No caso do “for”, o desvio é feito para o incremento, seguido pelo teste condicional e corpo do laço
- Esse comando deve ser evitado, pois dificulta a leitura e manutenção do código

# Comando goto

---

- Comando considerado dispensável para a programação estruturada
- Alguns programadores defendem o seu uso e outros rejeitam
- O fato é que com as três estruturas de laço vistas anteriormente (for, while, do-while), o “goto” não é necessário
- Suportado em algumas linguagens para manter a compatibilidade com o BASIC e FORTRAN

# Comando goto

---

```
int main()
{
    goto p2;
    printf("passou por aqui - p1 \n");

p2: printf("passou por aqui - p2 \n");

    system("pause");
}
```

# Vetores em C

---

- Variável para armazenar valores homogêneos e é unidimensional
- Declaração em C  
<tipo> <nome\_variável> [<tamanho>]

Ex:

```
int x[10];
```

- Como trabalhar com “string” em C?

# Vetores em C

---

- Um vetor do tipo “char” pode armazenar “string”
- Note que uma string sempre termina com o caracter *null* (“\0”)
- Exemplo em C

```
char nome[3] = "Ana";
```

```
char sobrenome[] = {'H','i','t','s'};
```

```
printf("%s,%d\n",nome,strlen(nome));
```

```
printf("%s,%d\n",sobrenome,strlen(sobrenome));
```

```
char sobrenome[] = {'H','i','t','s','\0'};
```

# Funções para trabalhar com strings

---

- Manipulação de strings:

`strlen();`

`strcat();`

`strcmp();`

`strcpy();`

- Entrada e saída

`gets();`

`puts();`

- Veja mais funções em:

[http://www.acm.uiuc.edu/webmonkeys/book/c\\_guide/](http://www.acm.uiuc.edu/webmonkeys/book/c_guide/)

# Exemplo com vetor

---

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 3
int main(int argc, char *argv[])
{
    int nota[MAX]; nota[0] = 10; nota[1] = 20; nota[2] = 30;
    //int nota[MAX] = {10,20,30};
    //int nota[] = {10,20,30};
    int media,x,acc=0;
    for(x=0; x<MAX;x++){
        acc += nota[x];
        printf("%d\n",nota[x]);
    }
    media = acc/MAX;
    printf("%d\n",media);
    system("PAUSE");
    return 0;
}
```

Note que em C o primeiro elemento do vetor é o índice [0]



# Exercício

---

- Escreva um programa que encontre o maior e o menor número de um vetor de inteiros de 30 elementos e mostre as suas respectivas posições;
  - Inicialize o vetor com números randômicos
    - Ex:

```
int x;
```

```
x = rand();
```

# Exercício

---

- Faça um programa que preencha um primeiro vetor com dez números inteiros e um segundo vetor com cinco números inteiros. O programa deverá mostrar uma lista dos números do primeiro vetor com seus respectivos divisores armazenados no segundo vetor, bem como suas posições
  - Para saber se um número inteiro é divisível por outro, deve-se calcular o resto da divisão
    - Exemplo em C

```
int x = 10, y = 2;  
printf("%d\n",x%y);
```

# Exercício

---

- Faça um programa que realize as seguintes operações com a string “ano2010”:
  - Converta o “ano” para caracteres maiúsculos
    - Saída: ANO2010
  - Insira um caracter de espaço em branco na quarta posição sem que haja perda de conteúdo
    - Saída: ANO 2010
  - Inverta o ordem da string
    - Saída: 0102ONA

# Exercício

---

- Escreva um programa para calcular o fatorial de um número inteiro digitado pelo usuário
- Escreva um programa para converter um valor informado pelo usuário da moeda no padrão USD para EU. O valor deverá ser armazenado no formato de string
  - Padrão:
    - USD 1,234.56 para Eu 1.234,56

# Solução: fatorial

---

```
int main()
{
    int num;
    long resposta;
    while(1)
    {
        printf("\n Digite o número: ", 163);
        scanf("%d", &num);
        resposta=1;
        while(num > 1)
            resposta *= num--;
        printf("O fatorial %c: %1d\n", 130, resposta);
        break;
    }
    system("pause");
}
```

# Referências

---

Ascencio AFG, Campos EAV. Fundamentos de programação de computadores. São Paulo : Pearson Prentice Hall, 2006. 385 p.

VICTORINE VIVIANE MIZRAHI, Treinamento em Linguagem C – Módulo 1 e Módulo 2, Makron Books, 1990.