

LÓGICA DE PROGRAMAÇÃO

PROF^a. M.Sc. JULIANA H Q BENACCHIO

- Ou comandos de **Iteração** ou **Laços**
- Permitem que um conjunto de instruções seja executado repetidamente:
 - enquanto uma determinada condição for verdadeira
 - ou até que uma determinada condição se torne falsa
 - ou repetir por um determinado número de vezes pré-definido

Acumuladores e Contadores

- Um **acumulador** é uma variável que ocorre em ambos os lados de uma atribuição e que, antes de ser usada pela primeira vez, é iniciada com um valor específico.
- Por exemplo,

 $a = 1;$

 $a = a + 2;$

A variável a passa a ter o valor 3.

Acumuladores e Contadores



INSTITUTO FEDERAL
PARANÁ

- É importante ressaltar que, se o valor inicial da variável que será utilizada como um **acumulador** não é definido pela primeira atribuição, não é possível determinar o seu valor após a execução da segunda atribuição.

Acumuladores e Contadores

- Pelo fato de operações com acumuladores serem tão comuns em programação, a linguagem C oferece um conjunto de **operadores aritméticos de atribuição** que permitem escrever expressões com acumuladores de uma forma mais compacta.

Operadores aritméticos de atribuição

- São combinações de operadores que simplificam as instruções. Dessa forma uma instrução escrita da forma:

$$x = x \text{ op } y;$$

- Pode ser reduzida obedecendo à sintaxe:

$$x \text{ op} = y;$$

Operadores aritméticos de atribuição

• Expressão Normal

`a = a + b;` →

`a = a - b;` →

`a = a * b;` →

`a = a / b;` →

`a = a % b;` →

Expressão Simplificada

`a+= b;`

`a-= b;`

`a*= b;`

`a/= b;`

`a%= b;`

Acumuladores e Contadores

- Um **contador** é um tipo de acumulador cujo valor aumenta, ou diminui, de 1 em 1.

| Operador | Operação | Operação |
|------------|-------------------------|-------------------|
| incremento | <code>i = i + 1;</code> | <code>i++;</code> |
| decremento | <code>i = i - 1;</code> | <code>i--;</code> |

Operadores de incremento e decremento

- Operadores de incremento e decremento podem ser usados na forma
 - prefixa (`++i` e `--i`)
 - posfixa (`i++` e `i--`)
- Na forma prefixa, o valor da variável é modificado e depois usado
- Na forma posfixa, o valor da variável é usado e depois modificado.

Operadores de incremento e decremento

. Por exemplo:

```
i = 3;
```

```
x = ++i;
```

prefixa

O valor de x será igual a 4 e i será igual a 4

```
i = 3;
```

```
x = i++;
```

posfixa

O valor de x será igual a 3 e i será igual a 4

Estruturas de Repetição



INSTITUTO FEDERAL
PARANÁ

- Na linguagem C existem três estruturas de laços:

- `while`

- `do-while`

- `for`

O Laço `while`

- Serve para executar um comando, repetidamente, **enquanto uma determinada condição for verdadeira**.
- Como a condição é avaliada antes de o comando ser executado, se ela for inicialmente falsa, o comando dentro da repetição jamais é executado.

O Laço `while`

- Execução:
 1. avalia a condição;
 2. se a condição for falsa, o comando é encerrado;
 3. se a condição for verdadeira, executa as instruções e retorna para o passo 1.

O Laço `while`

- Sua forma geral é:

```
while (condição)
{
    instrução1;
    instrução2;
}
```

O Laço `while`

- Por exemplo, para mostrar uma sequência de números de 1 a 10

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

O Laço `while`

- Por exemplo, para mostrar uma sequência de números de 1 a 10

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


O Laço `while`

- Para mostrar uma sequência de números de 1 a 10, de trás para frente

```
int i = 10;
while ( i > 0 )
{
    printf("%d", i);
    i--;
}
```

O Laço `while`

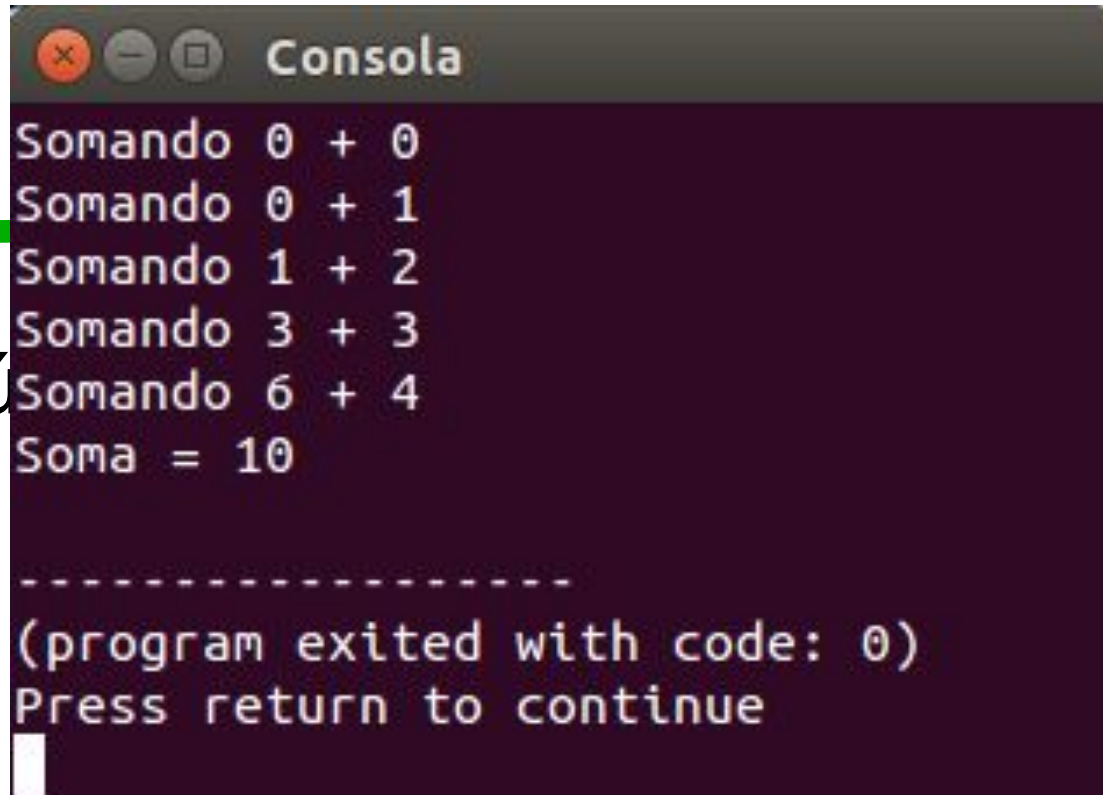
- Fazer a soma dos números enquanto ela for menor que 10

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

O Laço while

- Fazer a soma dos números menor que 10

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



```
Consola
Somando 0 + 0
Somando 0 + 1
Somando 1 + 2
Somando 3 + 3
Somando 6 + 4
Soma = 10

-----
(program exited with code: 0)
Press return to continue
```

O Laço `while`

- Fazer um programa que calcula a soma dos N primeiros números ($1+2+3+\dots+N$), sendo que N é um número digitado pelo usuário.

O Laço `while`

- Fazer um programa que calcula a soma dos N primeiros números ($1+2+3+\dots+N$), sendo que N é um número digitado pelo usuário.

```
int n, i = 1, soma = 0;
printf("Digite N: ");
scanf("%d", &n);
while(i <= n)
{
    soma = soma + i;
    i++;
}
printf("Soma = %d", soma);
```

O Laço `while`

- **Exercício:** Transformar o exemplo anterior para calcular o fatorial do número N digitado pelo usuário

$$5! = 1 \times 2 \times 3 \times 4 \times 5 = 120$$

- Relembrando: Por definição tanto $0!$, quanto $1!$ são iguais a 1

O Laço while

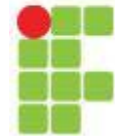
```
int n, i = 1, fat = 1;
printf("Digite N: ");
scanf("%d", &n);
while(i <= n)
{
    fat = fat * i;
    i++;
}
printf("Fatorial = %d", fat);
```

O Laço while

- Exemplo com caracteres:

```
char ch;  
printf("Digite uma letra\n");  
printf("Digite [s] para sair\n");  
while (ch != 's')  
{  
    scanf("%c", &ch);  
}
```


O Laço while

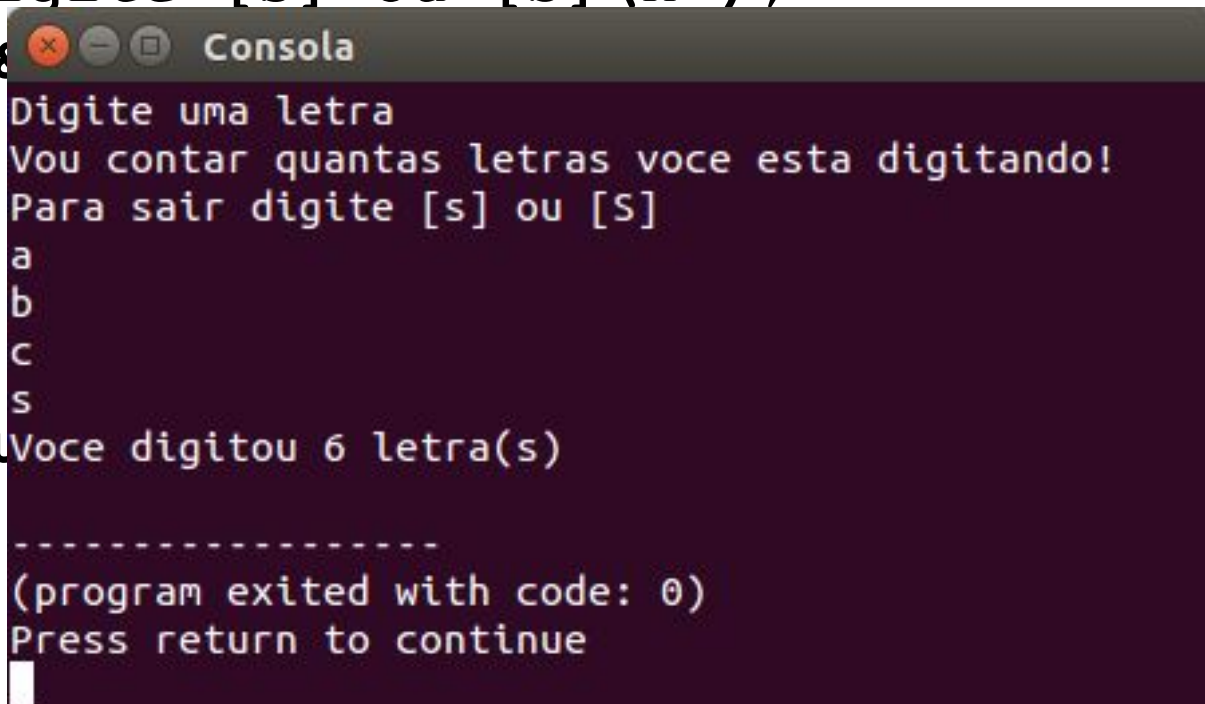


```
char ch;
int i = 0;
printf("Digite uma letra\n");
printf("Vou contar quantas letras voce esta
digitando!\n");
printf("Para sair digite [s] ou [S]\n");
while ((ch != 's') && (ch != 'S'))
{
    scanf("%c", &ch);
    i++;
}
printf("Voce digitou %d letra(s)", i);
```

O Laço while

```
char ch;  
int i = 0;  
printf("Digite uma letra\n");  
printf("Vou contar quantas letras voce esta  
digitando!\n");  
printf("Para sair digite [s] ou [S]\n");  
while ((ch != 's') &&  
{  
    scanf("%c", &ch);  
    i++;  
}  
printf("Voce digitou
```

ERRADO !!



```
Consola  
Digite uma letra  
Vou contar quantas letras voce esta digitando!  
Para sair digite [s] ou [S]  
a  
b  
c  
s  
Voce digitou 6 letra(s)  
-----  
(program exited with code: 0)  
Press return to continue
```

O Laço while



```
char ch;
int i = 0;
printf("Digite uma letra\n");
printf("Vou contar quantas letras voce esta
digitando!\n");
printf("Para sair digite [s] ou [S]\n");
while ((ch != 's') && (ch != 'S'))
{
    scanf("%c%c", &ch);
    i++;
}
printf("Voce digitou %d letra(s)", i-1);
```

O Laço while



```
char ch;
int i = 0;
printf("Digite uma letra\n");
printf("Vou contar quantas letras voce esta digitando!\n");
printf("Para sair digite [s] ou [S]");
while ((ch != 's') && (ch != 'S'))
{
    scanf("%c", &ch);
    i++;
}
printf("Voce digitou %d letra(s)", i);
```

```
Consola
Digite uma letra
Vou contar quantas letras voce esta digitando!
Para sair digite [s] ou [S]
a
b
c
s
Voce digitou 3 letra(s)
-----
(program exited with code: 0)
Press return to continue
```

O Laço `while`

- O formato `%*c` pode ser utilizado no `scanf()` para descartar o *enter* digitado ao final da entrada da resposta do usuário.

O Laço `while`

- Se a condição de parada é:
sair se o usuário digitar a letra S
minúscula OU maiúscula
- Por que a condição no `while` está com o
operador lógico E (`&&`) ??

```
printf("Para sair digite [s] ou [S]\n");  
while ((ch != 's') && (ch != 'S'))
```

O Laço **while**



- As operações lógicas no **while** devem ser analisadas de forma diferente

- O que seria feito no if como:

```
if (ch != 's' || ch != 'S')
```

- Para se conseguir o mesmo efeito no **while** deve ser:

```
while (ch != 's' && ch != 'S')
```

O Laço `while`

- O `while` irá executar a sequencia de instruções **enquanto a condição for verdadeira**
- As duas condições `(ch != 's ')` e `(ch != 'S ')` devem ser verdadeiras para o laço continuar sendo executado
- Portanto, enquanto o usuário digitar qualquer letra diferente de s ou S, a condições serão verdadeiras
- Mas vamos analisar as tabelas verdade nos outros casos...

O Laço `while`

Verdadeiro somente
no caso do usuário
digitar qualquer letra

| <u>(ch != 's')</u> | <u>(ch != 'S')</u> | <u>(ch != 's')</u> && <u>(ch != 'S')</u> |
|--------------------|--------------------|--|
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | F |

Verdadeiro também
para caso o usuário
digitar s ou S

| <u>(ch != 's')</u> | <u>(ch != 'S')</u> | <u>(ch != 's')</u> <u>(ch != 'S')</u> |
|--------------------|--------------------|--|
| V | V | V |
| V | F | V |
| F | V | V |
| F | F | F |

Exercícios



INSTITUTO FEDERAL
PARANÁ

- 1) Dadas as notas dos alunos de uma turma, informe a média da turma. O programa deve continuar lendo as notas da turma enquanto não for digitada uma nota negativa.

- 2) Dadas as idades dos pacientes de uma clínica, informe a idade daquele mais idoso. Considere que todas as idades são distintas e que o número de pacientes é informado pelo usuário, no momento inicial da execução do programa.