

Linguagem de Programação II

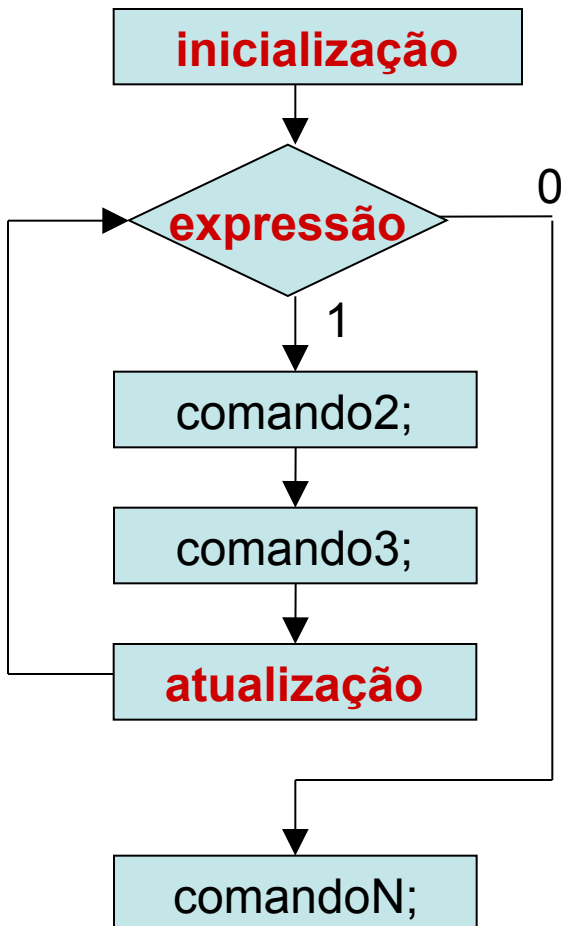
Prof. Mario Bessa

Aula 8

<http://mariobessa.info>

Laço “for”

- É uma simplificação do laço **while**, onde a inicialização da variável de controle, a expressão lógica e a atualização da variável são especificadas no próprio comando.



```
comando1;  
for(inicialização; expressão; atualização){  
    // bloco de comandos.  
    comando2;  
    comando3;  
    :  
}  
comandoN;
```

Laço “for” (Exemplo)

- **Problema:** somar **n** valores lidos da entrada padrão.

```
int n, i, num, soma=0;

printf("Entre com a quantidade: ");
scanf("%d",&n);
for(i=0; i<n; i++){
    scanf("%d",&num);
    soma += num; // soma = soma + num;
}
printf("Soma: %d\n",soma);
```

Laço “for” (Exemplo)

- **Problema:** Calcular o fatorial de um número.

```
int num, fat=1;

printf("Digite um número: ");
scanf("%d",&num);
for(int i=1;i<=num;i++)
    fat *= i;    // fat = fat * i;
}
printf("O fatorial de %d é %d",num,fat);
```

Comandos Repetitivos (laços)

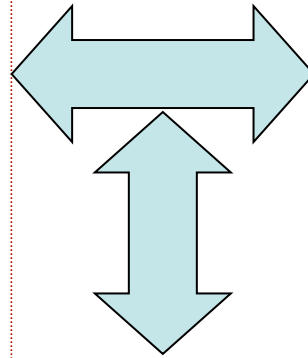
- Os comandos **break** e **continue**:
 - Podem ser usados no corpo de qualquer estrutura de laço C.
 - O **break** causa a saída imediata do laço e o controle passa para a próxima instrução após o laço.
 - O **continue** força a próxima iteração do laço e pula o código que estiver abaixo.

```
soma = 0;
for(i=0; i<n; i++){
    scanf("%d",&num);
    if(num<0) continue; // filtrar números negativos
    soma += num; // soma = soma + num;
}
```

Comandos Repetitivos (laços)

- **Equivalências:**

```
i=1;  
while(exp1){  
    comando1;  
    comando2;  
    i++;  
    ...  
}
```



```
for(i=1;exp1;i++){  
    comando1;  
    comando2;  
    ...  
}
```

```
i=1;  
do{  
    comando1;  
    comando2;  
    i++;  
    ...  
} while(exp1);
```

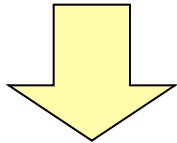
Laços aninhados

$$S = \sum_{i=1}^n \sum_{j=1}^m \frac{i^2 * j}{3^i (j * 3^i + i * 3^j)}$$

```
#include <stdio.h>
#include <math.h>
int main(){
    int i,j,m,n;
    float S=0.0,pi,pj,v;
    scanf("%d %d",&n,&m);
    for(i=1; i<=n; i++){
        for(j=1; j<=m; j++){
            pi = powf(3.0, i);
            pj = powf(3.0, j);
            v = (float) i * i * j;
            v /= pi*(j*pi + i*pj);
            S += v;    // S = S + v;
        }
    }
    printf("Soma: %f\n",S);
    return 0;
}
```

Laços aninhados

$$S = \sum_{i=1}^n \sum_{j=1}^m \frac{i^2 * j}{3^i (j * 3^i + i * 3^j)}$$



$$S = \sum_{i=1}^n \left(\frac{i^2}{3^i} \sum_{j=1}^m \frac{j}{j * 3^i + i * 3^j} \right)$$

```
#include <stdio.h>
#include <math.h>
int main(){
    int i,j,m,n;
    float S=0.0,Si,pi,pj;
    scanf("%d %d",&n,&m);
    for(i=1; i<=n; i++){
        Si = 0.0;
        pi = powf(3.0, i);
        for(j=1; j<=m; j++){
            pj = powf(3.0, j);
            Si += j / (j*pi + i*pj);
        }
        S += (Si*i*i) / pi;
    }
    printf("Soma: %f\n",S);
    return 0;
}
```


Laços aninhados

- **Problema:**

- Use o comando **for** para imprimir as seguintes sequências de números:

- (1, 1 2 3 4 5 6 7 8 9 10)

- (2, 1 2 3 4 5 6 7 8 9 10)

- (3, 1 2 3 4 5 6 7 8 9 10)

- (4, 1 2 3 4 5 6 7 8 9 10)

- ...

- (10, 1 2 3 4 5 6 7 8 9 10)

e assim sucessivamente, até que o primeiro número (antes da vírgula), também chegue a 10.

Laços aninhados

- **Problema:** sequência de números

```
#include <stdio.h>
int main(){
    for (int n=1;n<=10;n++) {
        printf("\n ( %d, ",n);
        for (int i=1;i<=10;i++)
            printf("%d ",i);
        printf(")");
    }
    return 0;
}
```

Laços aninhados

- **Problema:**
 - Use o comando **for** para mostrar uma pirâmide semelhante a abaixo, sendo que o maior valor da pirâmide é definido pelo usuário. Ex: n=9

```
9  8  7  6  5  4  3  2  1
8  7  6  5  4  3  2  1
7  6  5  4  3  2  1
6  5  4  3  2  1
5  4  3  2  1
4  3  2  1
3  2  1
2  1
1
```

Laços aninhados

- **Problema:** pirâmide

```
#include <stdio.h>
int main(){
    int n,i,j;
    scanf("%d",&n);
    for(i=n; i>0; i--){
        for(j=i; j>0; j--)
            printf(" %d ", j);
        printf("\n");
    }
    return 0;
}
```

Laços aninhados

- **Problema:**
 - Use o comando **for** para mostrar uma tabuada semelhante a abaixo:

TABUADA DO 2	TABUADA DO 3	TABUADA DO 4	TABUADA DO 5
2 x 1 = 2	3 x 1 = 3	4 x 1 = 4	5 x 1 = 5
2 x 2 = 4	3 x 2 = 6	4 x 2 = 8	5 x 2 = 10
2 x 3 = 6	3 x 3 = 9	4 x 3 = 12	5 x 3 = 15
2 x 4 = 8	3 x 4 = 12	4 x 4 = 16	5 x 4 = 20
2 x 5 = 10	3 x 5 = 15	4 x 5 = 20	5 x 5 = 25
2 x 6 = 12	3 x 6 = 18	4 x 6 = 24	5 x 6 = 30
2 x 7 = 14	3 x 7 = 21	4 x 7 = 28	5 x 7 = 35
2 x 8 = 16	3 x 8 = 24	4 x 8 = 32	5 x 8 = 40
2 x 9 = 18	3 x 9 = 27	4 x 9 = 36	5 x 9 = 45

TABUADA DO 6	TABUADA DO 7	TABUADA DO 8	TABUADA DO 9
6 x 1 = 6	7 x 1 = 7	8 x 1 = 8	9 x 1 = 9
6 x 2 = 12	7 x 2 = 14	8 x 2 = 16	9 x 2 = 18
6 x 3 = 18	7 x 3 = 21	8 x 3 = 24	9 x 3 = 27
6 x 4 = 24	7 x 4 = 28	8 x 4 = 32	9 x 4 = 36
6 x 5 = 30	7 x 5 = 35	8 x 5 = 40	9 x 5 = 45
6 x 6 = 36	7 x 6 = 42	8 x 6 = 48	9 x 6 = 54
6 x 7 = 42	7 x 7 = 49	8 x 7 = 56	9 x 7 = 63
6 x 8 = 48	7 x 8 = 56	8 x 8 = 64	9 x 8 = 72
6 x 9 = 54	7 x 9 = 63	8 x 9 = 72	9 x 9 = 81

Laços aninhados

- **Problema:** tabuada

```
#include <stdio.h>

int main(){
    int i,j,k;
    for(k=0; k<=1; k++){
        printf("\n");
        for(i=1; i<5; i++){
            printf(" TABUADA DO %d ",i+4*k+1);
            printf("\n");
            for(i=1; i<=9; i++){
                for(j=2+4*k; j<=5+4*k; j++){
                    printf(" %d x %d = %2d ", j, i, j*i);
                    printf("\n");
                }
            }
        }
        return 0;
    }
```

Laços aninhados

- **Problema:**
 - Use o comando **for** para mostrar a figura abaixo:

```
. # # # # # # # # # .  
. . # # # # # # # . .  
. . . # # # # # . . .  
. . . . # # # . . . .  
. . . . . # . . . . .  
. . . . . # . . . . .  
. . . . . # . . . . .  
. . . . . # # # . . . . .  
. . . . # # # # # . . . .  
. . . # # # # # # # . . .  
. . # # # # # # # # . .  
. # # # # # # # # # .
```

Laços aninhados

- **Problema:** pontos em 2D

```
#include <stdio.h>
int main(){
    int x,y,expr;

    for(y=10; y>=0; y--){
        for(x=0; x<=10; x++){
            expr = (x<y && y>10-x) || (x>y && y<10-x);
            if(expr) // expr==1
                printf(" # ");
            else // expr==0
                printf(" . ");
        }
        printf("\n");
    }
    return 0;
}
```


- **Problema:**

- Número de Armstrong: é um número de n dígitos que é igual a soma de cada um dos seus dígitos elevado a n -ésima potência.
- Exemplos:
 - $153 = 1^3 + 5^3 + 3^3$
 - $370 = 3^3 + 7^3 + 0^3$
 - $371 = 3^3 + 7^3 + 1^3$
 - $407 = 4^3 + 0^3 + 7^3$
- Elabore um programa em C para imprimir todos os números Armstrong entre 1 e 99999999.

```
#include <stdio.h>
#include <math.h>
int main(){
    int soma,temp,resto,n;
    for (int num=1; num<9999999; num++) {
        n=0;
        soma=0;
        temp = num;
        while(temp != 0){
            n++;
            temp = temp/10;
        }
        temp = num;
        while(temp != 0){
            resto = temp%10;
            soma = soma + pow(resto,n);
            temp = temp/10;
        }
        if ( num == soma )printf("%i ",num);
    }
    return 0;
}
```

Problemas

- Escrever um programa em C para gerar os n primeiros termos da sequência de Fibonacci 1,1,2,3,5,8,13,21...O valor de n será digitado pelo usuário. Usar comando **for**.
- A série de Ricci difere da série de Fibonacci porque os dois primeiros termos podem ser definidos pelo usuário. Escreva um programa em C que imprima os n primeiros termos da série de Ricci. O valor de n será digitado pelo usuário. Usar comando **for**.
- Escrever um programa em C que leia um valor inteiro n e teste se esse valor pertence à série de Fibonacci 1,1,2,3,5,8,13,21... O valor de n será digitado pelo usuário.
- Considere a série 1, 4, 4, 2, 5, 5, 3, 6, 6, 4, 7, 7, ...Escreva um programa em C que seja capaz de gerar os n termos dessa série. O valor de n será digitado pelo usuário.

Problemas

- Reescreva o exercício anterior, mas considere que o usuário irá digitar a posição do elemento pesquisado. Portanto, não é necessário escrever todos os elementos da série, mas apenas o elemento da posição pesquisada. Por exemplo, se o usuário digitar 11, o algoritmo deve imprimir apenas o valor 7 (uma vez que esse valor corresponde ao elemento da décima primeira posição).
- Escreva um algoritmo que calcule o número de notas que deve ser dado de troco para um pagamento efetuado. O algoritmo deve ler o valor pago em dinheiro e o valor do produto. Supor que o troco seja dado em notas de 50, 20, 10, 5, 2 e 1 real. O algoritmo deve usar uma quantidade mínima de notas. Por exemplo, se o usuário paga R\$100,00 por um produto de R\$ 17,00 receberá um troco de R\$83,00 com as seguintes notas: 1 nota de 50, 1 nota de 20, 1 nota de 10, 1 nota de 2 e 1 nota de 1.

Problemas: comandos de repetição

- Escrever um programa em C para gerar os n primeiros termos da sequência de Fibonacci 1,1,2,3,5,8,13,21...O valor de n será digitado pelo usuário. Usar comando **for**.

```
#include <stdio.h>
int main(){
    int x1=1,x2=1,x3,n;
    printf("\nEntre com o número de termos da sequência ");
    scanf("%i",&n);
    printf("\n%i %i ",x1,x2);
    for (int i=3; i<=n; i++) {
        x3=x1+x2;
        printf("%i ",x3);
        x1=x2;
        x2=x3;
    }
    return 0;
}
```

Problemas: comandos de repetição

- E se $n=0$. O que será impresso?

```
#include <stdio.h>
int main(){
    int x1=1,x2=1,x3,n;
    printf("\nEntre com o número de termos da sequência ");
    scanf("%i",&n);
    if (n<=0)
        printf("Valor inválido!!");
    else
        printf("\n%i %i ",x1,x2);
        for (int i=3; i<=n; i++) {
            x3=x1+x2;
            printf("%i ",x3);
            x1=x2;
            x2=x3;
        }
    return 0;
}
```

Problemas: comandos de repetição

- E se $n=1$ ou $n=2$. O que será impresso?

```
#include <stdio.h>
int main(){
    int x1=1,x2=1,x3,n;
    printf("\nEntre com o número de termos da sequência ");
    scanf("%i",&n);
    if (n<=0)
        printf("Valor inválido!!");
    else
        if (n==1)
            printf("1");
        else
            if (n==2)
                printf("1 2");
            else
                printf("\n%i %i ",x1,x2);
                for (int i=3; i<=n; i++) {
                    x3=x1+x2;
                    printf("%i ",x3);
                    x1=x2;
                    x2=x3;
                }
    return 0;
}
```

Problemas: comandos de repetição

- Escrever um programa em C para gerar os n primeiros termos da sequência de Fibonacci 1,1,2,3,5,8,13,21...O valor de n será digitado pelo usuário. Usar comando **while**.

```
#include <stdio.h>
int main(){
    int x1=1,x2=1,x3,n,i;
    printf("\nEntre com o número de termos da sequência ");
    scanf("%i",&n);
    if (n<=0)
        printf("Valor inválido!!");
    else
        if (n==1)
            printf("1");
        else
            if (n==2)
                printf("1 2");
            else{
                printf("\n%i %i ",x1,x2);
                i=3;
                while (i<=n) {
                    x3=x1+x2;
                    printf("%i ",x3);
                    x1=x2;
                    x2=x3;
                    i++;
                }
            }
    return 0;
}
```


Problemas: comandos de repetição

- Escrever um programa em C para gerar os n primeiros termos da sequência de Fibonacci 1,1,2,3,5,8,13,21...O valor de n será digitado pelo usuário. Usar comando **do...while**.

```
#include <stdio.h>
int main(){
    int x1=1,x2=1,x3,n,i;
    printf("\nEntre com o número de termos da sequência ");
    scanf("%i",&n);
    if (n<=0)
        printf("Valor inválido!!");
    else
        if (n==1)
            printf("1");
        else
            if (n==2)
                printf("1 2");
            else{
                printf("\n%i %i ",x1,x2);
                i=3;
                do{
                    x3=x1+x2;
                    printf("%i ",x3);
                    x1=x2;
                    x2=x3;
                    i++;
                }while (i<=n);
            }
    return 0;
}
```

Problemas: comandos de repetição

- A série de Ricci difere da série de Fibonacci porque os dois primeiros termos podem ser definidos pelo usuário. Escreva um programa em C que imprima os n primeiros termos da série de Ricci. O valor de n será digitado pelo usuário. Usar comando **for**.

```
#include <stdio.h>
int main(){
    int x1,x2,x3,n;
    printf("\nEntre com o número de termos da sequência ");
    scanf("%i",&n);
    printf("\nEntre com o primeiro termo da sequência ");
    scanf("%i",&x1);
    printf("\nEntre com o primeiro termo da sequência ");
    scanf("%i",&x2);
    printf("\n%i %i ",x1,x2);
    if (!(n==1 || n==2))
        for (int i=3; i<=n; i++) {
            x3=x1+x2;
            printf("%i ",x3);
            x1=x2;
            x2=x3;
        }
    return 0;
}
```

Problemas: comandos de repetição

- Escrever um programa em C que leia um valor inteiro n e teste se esse valor pertence à série de Fibonacci 1,1,2,3,5,8,13,21...O valor de n será digitado pelo usuário. Usar comando **while**.

```
#include <stdio.h>
int main(){
    int x1=1,x2=1,x3,n;
    printf("\nEntre com um número ");
    scanf("%i",&n);
    if (n<=0)
        printf("Valor inválido!!");
    else{
        while (n>x2){
            x3=x1+x2;
            x1=x2;
            x2=x3;
        }
        if (n==x2)
            printf("\nNúmero %i pertence à sequência",n);
        else
            printf("\nNúmero %i não pertence à sequência",n);
    }
    return 0;
}
```

Problemas: comandos de repetição

- Considere a série 1, 4, 4, 2, 5, 5, 3, 6, 6, 4, 7, 7, ...Escreva um programa em C que seja capaz de gerar os n termos dessa série. O valor de n será digitado pelo usuário. Usar comando **do..while**

Problemas: comandos de repetição

```
#include <stdio.h>
int main(){
    int n, posicao, prim, seg, terc;
    posicao=1;
    prim=1;
    seg=4;
    terc=4;
    printf("\nEntre com o valor de N ");
    scanf("%i",&n);
    do {
        if (n>=posicao) {
            printf("%i ",prim);
            prim++;
            posicao++;
        }
        if (n>=posicao) {
            printf("%i ",seg);
            seg++;
            posicao++;
        }
        if (n>=posicao) {
            printf("%i ",terc);
            terc++;
            posicao++;
        }
    } while (posicao<=n);
    return 0;
}
```

Problemas: comandos de repetição

- Reescreva o exercício anterior, mas considere que o usuário irá digitar a posição do elemento pesquisado. Portanto, não é necessário escrever todos os elementos da série, mas apenas o elemento da posição pesquisada. Usar comando **while**.

Problemas: comandos de repetição

```
#include <stdio.h>
int main(){
    int n, posicao=1, prim=1, seg=4, terc=4, achou=0;
    printf("\nEntre com a posição do elemento ");
    scanf("%i",&n);
    while (!achou) {
        if (n==posicao) {
            printf("%i ",prim);
            achou=1;
        }
        prim++;
        posicao++;
        if (n==posicao) {
            printf("%i ",seg);
            achou=1;
        }
        seg++;
        posicao++;
        if (n==posicao) {
            printf("%i ",terc);
            achou=1;
        }
        terc++;
        posicao++;
    }
    return 0;
}
```

Problema

- Escreva um programa em C que calcule o número de notas que deve ser dado de troco para um pagamento efetuado. O programa deve ler o valor pago em dinheiro e o valor do produto. Supor que o troco seja dado em notas de 50, 20, 10, 5, 2 e 1 real. O programa deve usar uma quantidade mínima de notas. Por exemplo, se o usuário paga R\$100,00 por um produto de R\$17,00 receberá um troco de R\$83,00 com as seguintes notas: 1 nota de 50, 1 nota de 20, 1 nota de 10, 1 nota de 2 e 1 nota de 1.

```
#include<stdio.h>
int main(){
    int dinheiro,preco,troco,quant;
    printf("Dinheiro: ");
    scanf("%d",&dinheiro);
    printf("Preço do produto: ");
    scanf("%d",&preco);
    quant=0;
    troco=dinheiro-preco;
    printf("Troco: %d",troco);
    if((troco/50) != 0){
        quant=troco/50;
        troco=troco-(quant*50);
        printf("\nA quantidade de notas de 50 : %d",quant);
    }
    if((troco/20) != 0){
        quant=troco/20;
        troco=troco-(quant*20);
        printf("\nA quantidade de notas de 20 : %d",quant);
    }
}
```

```
    if((troco/10) != 0){
        quant=troco/10;
        troco=troco-(quant*10);
        printf("\nA quantidade de notas de 10 : %d",quant);
    }
    if((troco/5) != 0){
        quant=troco/5;
        troco=troco-(quant*5);
        printf("\nA quantidade de notas de 5 : %d",quant);
    }
    if((troco/2) != 0){
        quant=troco/2;
        troco=troco-(quant*2);
        printf("\nA quantidade de notas de 2 : %d",quant);
    }
    if((troco/1) != 0){
        quant=troco/1;
        troco=troco-(quant*1);
        printf("\nA quantidade de notas de 1 : %d",quant);
    }
    return 0;
}
```