Laços encaixados

Disciplina de Programação de Computadores I
Universidade Federal de Ouro Preto

Agenda

- Laços encaixados
- Problemas com laços encaixados

Laços encaixados

- Um laço serve para iterar sobre os valores de uma variável, ou seja, executamos ações para cada possível valor de uma variável

- Quando precisamos iterar sobre os valores de 2 ou mais variáveis ao mesmo tempo, devemos utilizar laços dentro de laços, também conhecidos como laços encaixados

- Teremos tantos níveis de encaixe quantas forem as variáveis que devemos contar

Exemplo: Quadrado de asteriscos

- Como imprimir um quadrado formado por asteriscos na tela?

```
* * * *
* * * *
* * * *
* * * *
```

- Devemos imprimir 4 linhas sendo que cada linha deve possuir 4 asteriscos, ou seja, 4 colunas.

  Um laço para as linhas e outro laço para as colunas

Código: Quadrado (lado = 4 asteriscos)

```
int main (int argc, char const *argv[]){
	int linha, coluna;

	for(linha = 1; linha <= 4; ++linha){
		for(coluna = 1; coluna <= 4; ++coluna){
			printf(" * ");
		}
		printf("\n");
	}
	return 0;
}
```

linha	coluna
1	1
	2
	3
	4
2	1
	2
	3
	4
...	...

Código: Quadrado de asteriscos (lado variável)

```
int main (int argc, char const *argv[]){
    int linha, coluna, lados;

    printf("Quantos lados tem o quadrado?\n");
    scanf("%d", &lados);

    for(linha = 1; linha <= lados; ++linha) {
        for(coluna = 1; coluna <= lados; ++coluna) {
            printf(" * ");
        }
        printf("\n");
    }
    return 0;
}
```

linha	coluna
1	1
	2
	...
	lados
2	1
	2
	...
	lados
...	...

Exemplo: Triângulo retângulo de asteriscos

- Como imprimir um triângulo retângulo formado por asteriscos na tela?

```
*
* *
* * *
* * * *
```

- Devemos imprimir 4 linhas sendo que a primeira tem 1 asterisco e cada próxima linha possui um asterisco a mais que a anterior.

Código: Triângulo retângulo (base = 4 asteriscos)

```
int main (int argc, char const *argv[]) {
    int linha, coluna, max;

    max = 1;
    for(linha = 1; linha <= 4; ++linha){
        for(coluna = 1; coluna <= max; ++coluna){
            printf(" * ");
        }
        printf("\n");
        max ++;
    }
    return 0;
}
```

linha	max	coluna
1	1	1
2	2	1
		2
3	3	1
		2
		3
4	4	1
		2
		3
		4

Código: Triângulo retângulo (base = 4*) V2.0

```
int main (int argc, char const *argv[]) {
	int linha, coluna;

	for(linha = 1; linha <= 4; ++linha){
		for(coluna = 1; coluna <= linha; ++coluna){
			printf(" * ");
		}
		printf("\n");
	}
	return 0;
}
```

linha	coluna
1	1
2	1
	2
3	1
	2
	3
4	1
	2
	3
	4

Exemplo: Triângulo retângulo invertido

- Como imprimir um triângulo retângulo invertido formado por asteriscos na tela?

```
* * * *
* * *
* *
*
```

- Devemos imprimir 4 linhas sendo que a primeira tem 4 asterisco e cada próxima linha possui um asterisco a menos que a anterior.

Código: Triângulo retângulo invertido (base = 4*)

```
int main (int argc, char const *argv[]) {
    int lin, colu;

    for(lin = 1; lin <= 4; ++lin){
        for(colu = 1; colu <= 4 - lin + 1; ++colu){
            printf(" * ");
        }
        printf("\n");
    }
    return 0;
}
```

linha	coluna	4 - linha + 1
1	1	4
	2	
	3	
	4	
2	1	3
	2	
	3	
3	1	2
	2	
4	1	1

Arranjo com repetição

- Suponha dois dados, um azul e um branco, cada um com faces que contêm valores de 1 a 6.

- Imprima todas as combinações possíveis para os resultados de uma jogada com desses dados.

Se representarmos uma jogada por um par (a,b), em que a é o valor do dado azul e b é o valor do dado branco, devemos imprimir todos os pares ordenados de (1,1) até (6,6).

Código: Arranjo com repetição

```
int main(int argc, char const *argv[])
{
    for (int a = 1; a <= 6; ++a)
    {
        for (int b = 1; b <= 6; ++b)
        {
            printf("(%d, %d)\n", a, b);
        }
    }
    return 0;
}
```

a	b		a	b		a	b
1	1		3	1		5	1
	2			2			2
	3			3			3
	4			4			4
	5			5			5
	6			6			6
2	1		4	1		6	1
	2			2			2
	3			3			3
	4			4			4
	5			5			5
	6			6			6

Arranjo sem repetição

- Suponha dois dados, os dois de cor branca, cada um com faces que contêm apenas valores de 1 a 4.

- Imprima todas as combinações possíveis para os resultados de uma jogada com desses dados.

Se representarmos uma jogada por um par (a1,a2), em que a1 e a2 são os valores dos dados brancos, devemos imprimir todos os pares ordenados de (1,1) até (4,4), notando que (1,4) e (4,1) são a mesma jogada, já que ambos os dados são brancos.

Código: Arranjo sem repetição

```
int main(int argc, char const *argv[])
{
    for (int a1 = 1; a1 <= 4; ++a1)
    {
        for (int a2 = 1; a2 <= a1; ++a2)
        {
            printf("(%d, %d)\n", a1, a2);
        }
    }
    return 0;
}
```

a1	a2
1	1
2	1
	2
3	1
	2
	3
4	1
	2
	3
	4

Exemplo: ordenação de vetores

- Como ordenar um vetor de maneira crescente ou decrescente?

  Exemplo:

  Notas = [14, 24, 7, 12, 15, 3, 2, 18]

  Após ordenação:

  Notas = [2, 3, 7, 12, 14, 15, 18, 24]

Código: ordenar um vetor de forma crescente

```c
void ordena(int vetor[],int tam){
    int i,j,aux;
    for (i=0;i<tam;i++){
        for (j=i+1;j<tam;j++){
            if (vetor[i]>vetor[j]){
                aux=vetor[i];
                vetor[i]=vetor[j];
                vetor[j]=aux;
            }
        }
    }
}

int main(){
    int tam=15,vetor[tam],aux,i;
    for (i=0;i<tam;i++){
        printf("Digite o %d elemento: ",i+1);
        scanf("%d",&vetor[i]);
    }
    ordena(vetor,tam);

    printf ("Os elementos do vetor ordenados sao: ");
    for (i=0;i<tam;i++)
        printf("%d, ",vetor[i]);
    return 0;
}
```

Código: ordenar um vetor de forma crescente

```
void ordena(int vetor[],int tam){
    int i,j,aux;
    for (i=0;i<tam;i++){
        for (j=i+1;j<tam;j++){
            if (vetor[i]>vetor[j]){
                aux=vetor[i];
                vetor[i]=vetor[j];
                vetor[j]=aux;
            }
        }
    }
}
```

Exercícios

1. Escrever um programa que receba os valores de n produtos e imprima estes valores de forma decrescente.

2. Faça um programa que imprima os valores de uma tabuada de 1 a 9.

exemplo: 1 x 1 = 1
1 x 2 = 2
.
.
.
9 x 9 = 81

Referências Bibliográficas

- Material de aula da disciplina Algoritmos, UFJF: https://sites.google.com/site/algoritmosufjf
- Material de aula do Prof. Ricardo Anido, da UNICAMP: http://www.ic.unicamp.br/~ranido/mc102/
- Material de aula da Profa. Virgínia F. Mota: https://sites.google.com/site/virginiaferm/home/disciplinas
- DEITEL, P; DEITEL, H. *C How to Program*. 6a Ed. Pearson, 2010.

Agradecimentos

- Professores do Departamento de Ciência da Computação da UFJF que gentilmente permitiram a utilização das videoaulas elaboradas por eles.