

Algoritmos

Profº Ms Gustavo Molina

Prof. Ms Gustavo Molina



- Graduado em Sistemas de Informação pelo MACKENZIE.
- Licenciado em Matemática pela UNIP.
- Pós – Graduado em Plataforma de Desenvolvimento Web pelo CLARETIANO.
- Pós – Graduado em IA pela faculdade Serra Geral
- Pós – Graduado em Gestão e Governança de Tecnologia da Informação pela UNIP
- Mestre em Engenharia Elétrica pela FEI
- Doutorando em Ciências da Educação pela Ivy Enber Christian University

Aula 7 - Laços de Repetição

Laços de Repetição

- Laços de repetição são utilizados para executar um bloco de código uma determinada quantidade de vezes, ou até que uma condição seja satisfeita.
- No Java temos 4 laços de repetição: FOR, WHILE, DO-WHILE e FOR EACH.

FOR

- O FOR, em português PARA, é utilizado para executar um bloco de código uma determinada quantidade de vezes, baseado no valor de uma variável, geralmente ‘ i ’ :

```
public static void main(String[] args){  
    for(int i=1; i<10; i++){  
        System.out.println("Valor de i="+i);  
    }  
}
```

A saída deste código será:

Valor de i=1
Valor de i=2
Valor de i=3
Valor de i=4
Valor de i=5
Valor de i=6
Valor de i=7
Valor de i=8
Valor de i=9

FOR

- Observe o código com calma, a variável ‘ i ‘ é criada com valor = 1, depois é avaliada até seu valor ser < 10, ou seja, 9, e a última parte ‘i++’ quer dizer que o ‘ i ‘ vai ser somado de um em um:

```
public static void main(String[] args){  
    for(int i=1; i<10; i++){  
        System.out.println("Valor de i="+i);  
    }  
}
```

FOR

O FOR pode ser utilizado também de maneira decrescente:

```
public static void main(String[] args){  
    for(int i=10; i>0; i--){  
        System.out.println("Valor de i="+i);  
    }  
}
```

A saída deste código será:

Valor de i=10
Valor de i=9
Valor de i=8
Valor de i=7
Valor de i=6
Valor de i=5
Valor de i=4
Valor de i=3
Valor de i=2
Valor de i=1

FOR

O 'i' pode ser incrementado em diferentes valores, como por exemplo de 2 em 2:

```
public static void main(String[] args){  
    for(int i=1; i<=10; i+=2){  
        System.out.println("Valor de i="+i);  
    }  
}
```

A saída deste código será:

Valor de i=1
Valor de i=3
Valor de i=5
Valor de i=7
Valor de i=9

*Observe que a contagem é de 2 em 2, não de 1 em 1.

FOR

- Dentro do FOR podemos colocar qualquer tipo de código, inclusive blocos IF:

```
public static void main(String[] args){  
    for(int i=1; i<=10; i++){  
        if(i%2==0){  
            System.out.println(i+" é um número par");  
        }else{  
            System.out.println(i+" é um número ímpar");  
        }  
    }  
}
```

FOR

- Saída do código do slide anterior:

```
1 é um número ímpar
2 é um número par
3 é um número ímpar
4 é um número par
5 é um número ímpar
6 é um número par
7 é um número ímpar
8 é um número par
9 é um número ímpar
10 é um número par
```

FOR

- Ao executar qualquer laço de repetição, podemos sair dele executando o comando BREAK e podemos pular uma volta com CONTINUE :

```
public static void main(String[] args){  
    for(int i=1; i<=10; i++){  
        if(i==5){  
            continue;  
        }  
  
        if(i==8){  
            break;  
        }  
        System.out.println("i="+i);  
    }  
}
```

FOR

- Saída do código do slide anterior:

i=1

i=2

i=3

i=4

i=6

i=7

FOR

Também podemos utilizar um FOR dentro de outro FOR:

```
public static void main(String[] args){
    for(int i=1; i<=10; i++){
        if(i%2==0){
            System.out.println("O número " + i + " é par. Pares antes: ");
            for(int j=1; j<i; j++){
                if(j%2==0){
                    System.out.print(j + " ");
                }
            }
            System.out.println();
        }else{
            System.out.println("O número " + i + " é ímpar");
        }
    }
}
```

FOR

- Saída do código do slide anterior:

O número 1 é ímpar

O número 2 é par e antes dele tem os seguintes pares:

O número 3 é ímpar

O número 4 é par e antes dele tem os seguintes pares:

2

O número 5 é ímpar

O número 6 é par e antes dele tem os seguintes pares:

2 4

O número 7 é ímpar

O número 8 é par e antes dele tem os seguintes pares:

2 4 6

O número 9 é ímpar

O número 10 é par e antes dele tem os seguintes pares:

2 4 6 8

Exercício

- Escreva um código Java para imprimir de 1 a 100 em ordem crescente, informando se o número é par, ímpar, múltiplo de 3, de 4 ou de 5.
- Se o número for múltiplo de 5, escreva outro FOR dentro do IF para imprimir todos os números múltiplos de 5 antes dele em ordem decrescente.
- EXTRA: verifique se os números são PRIMOS!

WHILE

O laço de repetição WHILE, em português ENQUANTO, é executado enquanto uma condição for verdadeira:

```
public static void main(String[] args){  
    int i = 0;  
    while(i<10){  
        System.out.println("i=" + i++);  
    }  
}
```

WHILE

- A saída do código anterior é:

```
i=0  
i=1  
i=2  
i=3  
i=4  
i=5  
i=6  
i=7  
i=8  
i=9
```

*Observe que o 10 não é impresso!

WHILE

- Um erro muito comum quando se utiliza o WHILE é não atualizar a variável que está sendo verificada, o que causa um laço infinito e o programa ‘trava’.
- Por isso o ‘i’ no código anterior está sendo incrementado de um em um DENTRO DO LAÇO!

WHILE

- O WHILE também pode ser utilizado para executar laços dos quais não sabemos o número de repetições necessárias, como por exemplo um MENU:

```
public static void main(String[] args){  
    Scanner leia = new Scanner(System.in);  
    int op=0;  
    while(op!=3){  
        System.out.println("1-START GAME");  
        System.out.println("2-LOAD GAME");  
        System.out.println("3-EXIT GAME");  
    }  
}
```

WHILE

- Saída do slide anterior para entradas ‘1’, ‘2’ e ‘3’:

```
1-START GAME  
2-LOAD GAME  
3-EXIT GAME
```

1

```
1-START GAME  
2-LOAD GAME  
3-EXIT GAME
```

2

```
1-START GAME  
2-LOAD GAME  
3-EXIT GAME
```

3

*Só quando se digita 3 é que sai do programa.

WHILE

Um outro exemplo do uso do WHILE seria para calcular a média de uma turma, da qual não sabemos ainda o número de alunos:

```
public static void main(String[] args){
    Scanner leia = new Scanner(System.in);
    int numeroAlunos=0;
    double mediaTurma=0;
    double mediaAluno=0;
    while(mediaAluno != -1){
        System.out.println("Digite a média do aluno:");
        mediaAluno = leia.nextDouble();
        if(mediaAluno != -1){
            numeroAlunos++;
            mediaTurma += mediaAluno;
        }
    }
    mediaTurma = mediaTurma / numeroAlunos;
    System.out.println("Média da turma = " + mediaTurma);
}
```

WHILE

- Saída do código anterior:

```
Digite a média do aluno:  
5  
Digite a média do aluno:  
5  
Digite a média do aluno:  
6  
Digite a média do aluno:  
6  
Digite a média do aluno:  
-1  
Média da turma = 5.5
```

*Observe que foi definido para sair do laço caso a média digitada fosse -1, pois não é uma nota válida

Números Aleatórios

- Em Jogos se utiliza muito a geração aleatória de números para criar objetos em posições aleatórias no cenário ou, mais recentemente, gerar cartas em LOOT.
- Em Java podemos utilizar o seguinte código:
 - Random ale = new Random();
 - int aleatório = 1+ale.nextInt(100);
- Este código gerará um número aleatório entre 1 e 100.

Exercícios

- Faça um jogo no qual será gerado um número aleatório entre 1 e 10000.
- O usuário poderá chutar até 20 vezes.
- Caso ele chute um número menor que o gerado aleatoriamente, imprima “é um número MAIOR”, se ele digitar um maior imprima “é um número MENOR”.
- Quando ele acertar, uma mensagem de PARABÉNS deve ser impressa
- Caso ele não acerte em 20 tentativas imprima “Infelizmente você perdeu!”

DO-WHILE

- O laço de repetição DO-WHILE executa a mesma função do WHILE com uma única diferença: sempre entra no laço

```
public static void main(String[] args){  
    int i = 1;  
    do{  
        System.out.println("i="+ i++);  
    } while(i<=10);  
}
```

- A condição é avaliada DEPOIS do laço
- Nesse caso vai imprimir de 1 a 10

DO-WHILE

Vejamos claramente a diferença entre DO-WHILE e WHILE:

```
public static void main(String[] args){
    int i = 1;
    do{
        System.out.println("i="+ i);
    } while(i != 1);
}

public static void main(String[] args){
    int i = 1;
    while(i!=1){
        System.out.println("i="+ i);
    }
}
```

- O do-while vai imprimir 1 e o while não!

DO-WHILE

- Por isso o DO-WHILE é utilizado quando desejamos entrar pelo menos uma vez no laço de repetição.
- Vejamos agora um código para imprimir um tabuleiro de xadrez, onde B são as casas brancas e P as casas pretas:

DO-WHILE

- Tabuleiro de Xadrez

```
public static void main(String[] args){  
    int i = 1;  
    int j = 1;  
    do{  
        do{  
            if((i+j)%2==0){  
                System.out.print("P "); j++;  
            }else{  
                System.out.print("B "); j++;  
            }  
        }while(j <=8 );  
    System.out.println();  
    j=1;  
    i++;  
    }while(i <=8 );
```

DO-WHILE

- Saída do código do slide anterior:

```
P B P B P B P B P B  
B P B P B P B P B P  
P B P B P B P B P B  
B P B P B P B P B P  
P B P B P B P B P B  
B P B P B P B P B P  
P B P B P B P B P B  
B P B P B P B P B P
```

Exercícios

- Tendo em vista o tabuleiro de xadrez, imprima, utilizando dois DO-WHILE, o tabuleiro e as posições das peças PRETAS:
 - T=Torre
 - C=Cavalo
 - b=Bispo
 - r=Rainha
 - R=Rei
 - p=Peão
 - P=casa preta, B=branca



Perguntas?