

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by two horizontal dark gray bars, one at the top and one at the bottom.

ORACLE

Academy

Java Fundamentals

5-1

Scanner e demonstrações condicionais

ORACLE
Academy



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Objetivos

- Esta aula abrange os seguintes tópicos:
 - Usar Scanner para entrada do usuário durante a execução do programa
 - Usar lógica e instruções if-else
 - Aplicar lógica e instruções de switch no código Java
 - Usar quebra e padrão de forma efetiva em uma instrução de switch
 - Usar o operador ternário



Solicitando a Entrada do Usuário: Scanner

- A entrada do teclado usando um Scanner requer a seguinte instrução de importação:

```
import java.util.Scanner;
```

- A solicitação do usuário pode ser feita com um simples código que aparecerá na tela da console, onde o usuário pode inserir sua entrada

```
System.out.println("Write instructions for user here.");
```

Se não estiver familiarizado com programação, você às vezes se esquecerá disso e se perguntará por que o programa está “parado” quando, na verdade, ele está apenas aguardando a entrada do usuário.

Scanner

- Para ler a entrada que o usuário inseriu, use o Scanner do objeto Java
- Para inicializar um Scanner, escreva:

```
Scanner in = new Scanner(System.in);
```

O scanner geralmente é chamado "in" que é a abreviatura de "input" (entrada)

System.in é usado pelo scanner para ler a entrada dos usuários da tela da console.

Tecnicamente, System.in é um arquivo que está associado ao telado ou à "entrada padrão". O digitalizador pode ser usado com objetos de arquivo para ler em um arquivo de texto.

Por que Usar o Scanner?



- O scanner facilita a leitura da entrada do usuário porque ela já tem métodos que executam esta tarefa bem
- O método do Scanner `next()` lê a entrada do usuário como uma String e retorna essa String
- Esta linha do código:
 - Cria uma nova string chamada entrada
 - Digitaliza a string que o usuário informou na console de saída usando o scanner chamado
 - Define a entrada igual à string que foi lida pelo scanner

```
String input = in.next();
```

Método nextInt() do Scanner

- O método do Scanner nextInt() lê a entrada do usuário como um inteiro e retorna esse inteiro
- Esta linha do código cria um novo inteiro chamado resposta

```
import java.util.Scanner;
public class InputExample{
    public static void main(String[] args){
        Scanner in = new Scanner(System.in);
        System.out.println("Enter your name:");
        String name = in.next(); //lê o texto até um espaço
        System.out.println("Enter a number:");
        int answer = in.nextInt(); //lê um valor inteiro
        System.out.println(name + ", the number you entered is: "
                           + answer);
    } //fim do método main
} //fim da classe InputExample
```

ORACLE
Academy

JF 5-1
Scanner e demonstrações condicionais

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

7

Como programador, você precisará escrever um código que lidará com situações em que tipos de dados inválidos são inseridos. Por exemplo, “vinte” quando o esperado seria o valor inteiro 20. Os programas profissionais têm métodos que verificam se há dados inválidos óbvios e não óbvios. Às vezes, isso é chamado de “blindar” o código.

Métodos Mais Úteis do Scanner

Método	O Que Ele Faz	Quando Usar
nextInt()	Semelhante a next(), esta função lê a entrada do usuário e retorna seu valor do inteiro.	Quando você solicita ao usuário um valor de inteiro e deseja ler a entrada do usuário como um inteiro, em vez de uma string.
hasNext()	Retorna verdadeiro, se o scanner tiver outra entrada, caso contrário, falso.	Quando quiser saber se há mais entrada para o scanner ler.
close()	Fecha o scanner.	Quando terminar de ler a entrada, é melhor fechar o scanner, principalmente ao ler a entrada da tela da console. Isso mantém o programa em execução continuamente. O scanner pode esperar mais entrada, se nunca foi fechado.

hasNext() e close() são usados para arquivos de leitura.

O não fechamento de objetos de Digitalizador ou outros arquivos pode causar a imobilização de recursos ou até mesmo danos em arquivos. Sempre feche objetos de Digitalizador assim que possível.

Operadores Relacionais



- O Java tem seis operadores relacionais usados para testar valores primitivos ou numéricos literais
- Os operadores relacionais são usados para avaliar as condições de loop e if-else

Operador Relacional	Definição
>	Maior que
>=	Maior que ou igual a
==	Igual a
<	Menor que
<=	Menor que ou igual a
!=	Não é igual a

Em Java, um erro de sintaxe comum é usar = quando se pretendia usar ==. Você deve ler = não como “é igual a”, mas como “é atribuído”.

Para obter mais informações sobre esses operadores e os exibidos no slide 12, verifique a tabela de precedência em <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/operators.html>

Exemplo de Operadores Relacionais

- Os valores são testados no lado do operador e um valor verdadeiro ou falso é retornado
- Este valor pode ser armazenado ou usado como parte de uma estrutura de controle para controlar o fluxo do programa
- Neste exemplo, a variável `madeHonorRoll` é designada a um valor verdadeiro quando a nota da expressão `>= 88` é avaliada como verdadeira

```
int grade = 99;  
boolean madeHonorRoll = grade >= 88;  
if (madeHonorRoll)  
    System.out.println("You made the Honor Roll.");
```

Exemplo de Operadores Relacionais

- O mesmo exemplo pode ser avaliado com o uso da variável booleana
- Porém, a nota da expressão ≥ 88 é avaliada como verdadeira ou falsa, dependendo do valor designado à nota
- Os valores booleanos são necessários como uma condição em um loop ou instrução if-else

```
int grade = 99;  
  
if(grade >= 88)  
    System.out.println("You made the Honor Roll.");
```

Operadores Lógicos

- O Java tem três operadores lógicos usados para combinar expressões booleanas em testes complexos

Operador Lógico	Significado
&&	E
	OU
!	NÃO

Exemplo 1 de Operadores Lógicos

- Neste exemplo, a frase "Você está qualificado para a bolsa" será impressa, se as duas condições forem verdadeiras
- Para que a mensagem seja impressa, madeHonorRoll deve ser verdadeiro e o numberDaysAbsent deve ser igual a zero

```
int numberDaysAbsent = 0;
int grade = 99;
boolean madeHonorRoll = grade >= 88;

if(madeHonorRoll && numberDaysAbsent==0)
    System.out.println("Você está qualificado para a bolsa.");
```

Exemplo 2 de Operadores Lógicos

- Descreve os resultados de cada um dos seguintes segmentos do código Java

```
double grade=65;
int numDaysAbsent=2;
boolean madeHonorRoll = grade >= 88;
if(!madeHonorRoll && numDaysAbsent<3)
    System.out.println("Você está qualificado para ajuda "
                        + "gratuita do tutorial.");
if(grade > 70 && numDaysAbsent < 5)
    System.out.println("You may try out for the sports team.");
```

Solução do Exemplo 2 de Operadores Lógicos

- Resultados de cada um dos seguintes segmentos do código Java:
 - A frase "Você está qualificado para a ajuda gratuita do tutorial" é impressa na tela
 - Porém, a frase, "Você pode tentar a equipe de esportes" não é impressa, pois a nota do aluno não está acima de 70

```
double grade=65;
int numDaysAbsent=2;
boolean madeHonorRoll = grade >= 88;
if(!madeHonorRoll && numDaysAbsent<3)
    System.out.println("Você está qualificado para ajuda "
                        + "gratuita do tutorial.");
if(grade > 70 && numDaysAbsent < 5)
    System.out.println("You may try out for the sports team.");
```

Sintaxe das Instruções if-else

- Para criar uma instrução if-else, lembre-se das seguintes regras:
 - Uma instrução if-else precisa de uma condição ou método que é testado como verdadeiro/falso
 - Por exemplo:

```
if (y > 17)
if (x == 5)
if (s1.equals(s2))
```



Você já passou pela instrução if-else nas seções do Alice e do Greenfoot.

Sintaxe das Instruções if-else

- Do mesmo modo, uma instrução else if opcional pode ser testada, por exemplo:

```
if(y > 17){  
    System.out.println("y > 17");  
else if(y == 17)  
    System.out.println("y == 17");
```

- A instrução else opcional cuidará de cada uma das outras possibilidades

```
if(y >= 17){  
    System.out.println("y >= 17");  
else if(y == 7)  
    System.out.println("y >= 17");  
else  
    System.out.println("y < 17");
```

Instruções if-else com o Tipo de Dados do caractere

```
import java.util.Scanner;
public class Calculator{
    public static void main(String[] args){
        Scanner in = new Scanner(System.in);
        int answer = 0;
        System.out.println("Enter a number: ");
        int num1 = in.nextInt();
        System.out.println("Enter another number: ");
        int num2 = in.nextInt();
        System.out.println("Enter the operand(* / % + -): ");
        char input = in.next().charAt(0);
        if( input == '*' )
            answer = num1 * num2;
        else if( input == '/' )
            answer = num1 / num2;
        else if( input == '%' )
            answer = num1%num2;
        else if( input == '+' )
            answer = num1 + num2;
        else if( input == '-' )
            answer = num1 - num2;
        else
            System.out.println("Invalid Command");
        System.out.println("The result is: " + answer);
    } //fim do método main
} //fim da classe Calculator
```

ORACLE
Academy

JF 5-1
Scanner e demonstrações condicionais

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

18

Se houver mais de uma instrução para if ou else, elas deverão ser inseridas entre {chaves}. A seguir vemos um erro de lógica comum:

```
if( false )
    System.out.println("I never print");
    System.out.println("I always print");
```

Instruções if-else com o Tipo de Dados do inteiro

- Pode ser útil usar print, em vez de println, ao solicitar uma entrada do usuário. Mantém o cursor na mesma linha:

```
import java.util.Scanner;
public class ValueChecker{
    public static void main(String[] args){
        Scanner in = new Scanner(System.in);
        int value = 0;
        System.out.print("Enter a number:");
        value = in.nextInt();
        if( value == 7)
            System.out.println("That's lucky!");
        else if( value == 13)
            System.out.println("That's unlucky!");
        else
            System.out.println("That is neither lucky nor unlucky!");
    } //fim do método main
} //fim da classe ValueChecker
```

Instruções if-else com o Tipo de Dados da String

- `nextLine()` lê todo o texto, independentemente de espaços

```
import java.util.Scanner;
public class StringChecker{
    public static void main(String[] args){
        Scanner in = new Scanner(System.in);
        String name = "";
        System.out.print("Enter your name:");
        name = in.nextLine();
        if(name.equals("Elvis"))
            System.out.println("You are the king of Rock and"
                               + " Roll");
        else if(name.equals("Michael Jackson"))
            System.out.println("You are the king of pop!");
        else
            System.out.println("You are not the king!");
    } //fim do método main
} //fim da classe StringChecker
```

AlterarInstrução

- Como no exemplo if-else anterior, considere um programa que usa duas entradas do inteiro de um usuário e executa uma operação matemática especificada
- Para suportar diferentes operadores é necessário um teste para ver se a entrada era uma das seguintes:
- * / + - %
- Como você verifica o que o usuário digitou?

A Instrução Switch Altera o Fluxo do Programa

- Uma instrução switch é outra forma de alterar o fluxo do programa, dependendo do valor de entrada

```
import java.util.Scanner;
public class Calculator{
    public static void main(String[] args){
        Scanner in = new Scanner(System.in);
        int answer = 0;
        System.out.print("Enter a number: ");
        int num1 = in.nextInt();
        System.out.print("Enter another number: ");
        int num2 = in.nextInt();
        System.out.println("Enter the operand: ");
        char input = in.next().charAt(0);
        ...
    }
}
```

A Instrução Switch Altera o Fluxo do Programa

```
...
    switch (input) {
        case '*' : answer = num1 * num2;
                    break;
        case '/' : answer = num1 / num2;
                    break;
        case '%' : answer = num1 % num2;
                    break;
        case '+' : answer = num1 + num2;
                    break;
        case '-' : answer = num1 - num2;
                    break;
        default: System.out.println("Invalid Command.");
    } //fim switch
    System.out.println("The result is: " + answer);
} //fim do método main
} //fim da classe Calculator
```

ORACLE
Academy

JF 5-1
Scanner e demonstrações condicionais

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

23

O sinal de dois-pontos (:) padrão, como o else opcional em uma instrução if-else, também é opcional. Lembre-se de que você usa dois-pontos (:) para as instruções de caso, e não ponto-e-vírgula (;) que representa o fim de uma instrução.

É preferível usar switch, em vez de if-else aninhado, porque é mais fácil de ler. No entanto, o switch não funciona bem com intervalos:

```
if(age >=13 && <=19)
    System.out.println("Teenager");
```

em oposição a:

```
switch(age)
case 13:
case 14:
case 15:
case 16:
case 17:
case 18:
case 19:
    System.out.println("Teenager");
    break;
```

Palavras-chave da Instrução Switch

- Uma instrução switch usa 3 palavras-chaves: switch, case e default
 - switch:
 - especifica qual variável para testar o valor
 - case:
 - compara o valor da variável switch
 - default:
 - quando a entrada não corresponder a nenhum dos casos, o compilador escolhe a ação padrão (como else em uma lista de instruções if)

Informações Adicionais sobre Instruções Switch

- Após cada case, inclua a palavra-chave break
- Se não for incluída, o código "falhará" e executará cada caso, até break ser encontrada
- No Java SE 7 e posterior, você pode usar um objeto String na expressão da instrução switch

Exemplo 1 da Instrução Switch

- Este exemplo mostra como o procedimento de "falha" funciona Para participação em vendas, as participações mais vendidas, os melhores preços que um vendedor obtém para essas vendas

Exemplo 1 da Instrução Switch

```
import java.util.Scanner;
public class SalesWinners {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        System.out.println("How many memberships did you sell?");
        int sales = in.nextInt();

        switch(sales){
            case 6: System.out.println("You win $1000");
            case 5: System.out.println("You win a Samsung Galaxy");
            case 4: System.out.println("You win Laptop");
            case 3: System.out.println("You win iPod");
            case 2: System.out.println("You win Stapler");
            case 1: System.out.println("You win Staple Remover");
                    break;
            default: System.out.println("No Gift");
        } //fim switch
    } //fim do método main
} //fim da classe SalesWinners
```

Exemplo 2 da Instrução Switch

- Um aluno do 9º ano do ensino fundamental II é considerado um principiante, os alunos do 10º ano são estudantes de segundo ano da universidade, etc

```
import java.util.Scanner;
public class ClassYear {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("What grade are you in?");
        int grade = in.nextInt();
        switch(grade) {
            case 9: System.out.println("You are a freshman");
                    break;
            case 10: System.out.println("You are a sophomore");
                    break;
            case 11: System.out.println("You are a junior");
                    break;
            case 12: System.out.println("You are a senior");
                    break;
            default: System.out.println("Invalid grade");
        } //fim switch
    } //fim do método main
} //fim da classe ClassYear
```

Exemplo 3 da Instrução Switch

- Dado um mês e um ano, o número de dias do mês são calculados
- Incentive os alunos a pesquisar as regras de "Ano Bissexto"

```
import java.util.Scanner;  
public class LeapYearCalculator {  
    public static void main(String[] args) {  
        Scanner in = new Scanner(System.in);  
        System.out.println("Enter the month");  
        int month = in.nextInt();  
        System.out.println("Enter the year");  
        int year = in.nextInt();  
        ...  
    }  
}
```

Exemplo 3 da Instrução Switch

```
...  
    switch (month) {  
        case 4:  
        case 6:  
        case 9:  
        case 11: System.out.println("That month has 30 days");  
            break;  
        case 1:  
        case 3:  
        case 5:  
        case 7:  
        case 8:  
        case 10:  
        case 12: System.out.println("That month has 31 days");  
            break;  
    }  
...
```

caso 2: leva em conta as regras de ano bissexto implementadas com o calendário gregoriano. Um ano bissexto a cada quatro anos produz muitos dias de ano bissexto a cada 400 anos. Dessa forma, embora 2000 tenha sido um ano bissexto, 1800 e 1900 não foram e 2100 não será.

Para se divertir, acesse <http://www.timeanddate.com> e digite o ano 1752 e o mês de setembro. Foi nesse ano que o Reino Unido mudou do calendário juliano para o gregoriano. Os dias ausentes foram para alinhar as estações no calendário. Outros países adotaram o calendário gregoriano em anos diferentes, sendo que a Grécia foi o último país a mudar em 1923.

Exemplo 3 da Instrução Switch

```
...  
    case 2:    if((year % 4 == 0 && year % 100 != 0) ||  
                year % 400 == 0)  
                System.out.println("That month has 29 days");  
            else  
                System.out.println("That month has 28 days");  
            break;  
    default:   System.out.println("Invalid Input");  
} //fim switch  
} //fim do método main  
} //fim da classe LeapYearCalculator
```

caso 2: leva em conta as regras de ano bissexto implementadas com o calendário gregoriano. Um ano bissexto a cada quatro anos produz muitos dias de ano bissexto a cada 400 anos. Dessa forma, embora 2000 tenha sido um ano bissexto, 1800 e 1900 não foram e 2100 não será.

Para se divertir, acesse <http://www.timeanddate.com> e digite o ano 1752 e o mês de setembro. Foi nesse ano que o Reino Unido mudou do calendário juliano para o gregoriano. Os dias ausentes foram para alinhar as estações no calendário. Outros países adotaram o calendário gregoriano em anos diferentes, sendo que a Grécia foi o último país a mudar em 1923.

Expressões switch

- Desde o Java 14, foi incluída uma expressão switch na linguagem Java
- O rótulo switch agora é escrito como: `case value -> code;`
- Se houver correspondência com o rótulo, somente o código à direita do rótulo será executado
- Depois de cada caso, não é obrigatório usar a palavra-chave `break`
- Os casos de uma expressão switch devem ser exaustivos: todos os valores possíveis devem ter um rótulo switch correspondente
- Em geral, é obrigatório um padrão

Expressões switch — Exemplo 1

- Você pode reescrever o exemplo de switch 2 usando os novos rótulos de expressão de switch
- Compare as duas maneiras para analisar as diferenças

```
import java.util.Scanner;
public class ClassYear {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("What grade are you in?");
        int grade = in.nextInt();
        switch(grade) {
            case 9 -> System.out.println("You are a freshman");
            case 10 -> System.out.println("You are a sophomore");
            case 11 -> System.out.println("You are a junior");
            case 12 -> System.out.println("You are a senior");
            default -> System.out.println("Invalid grade");
        } //fim de switch
    } //fim do método main
} //fim da classe ClassYear
```

Operador Ternário

- O operador ternário (?:) que é usado no Java para criar uma versão mais curta de uma instrução if-else
- No exemplo a seguir, há três parâmetros que usam este operador
 - O primeiro é o teste booleano (c>9)
 - O segundo (6) é o valor para retornar se o teste é verdadeiro
 - O terceiro (7) é o valor para retornar se o teste for falso
 - Geralmente é usado como parte de uma designação

```
int x = c > 9 ? 6 : 7;  
//Se c for maior que 9, x será 6; caso contrário, x será 7
```

O operador ternário, às vezes, é chamado de operador condicional. É o único operador em Java a ter três operandos.

Exemplo de Operador Ternário

- Aqui, uma instrução if-else é usada para verificar a igualdade da String

```
String s1 = "Hello";  
String s2 = "Goodbye";  
if(s1.equals(s2))  
    System.out.println("Yes");  
else  
    System.out.println("No");
```

- Um resultado semelhante é obtido usando o operador ternário

```
String s1 = "Hello";  
String s2 = "Goodbye";  
String answer = s1.equals(s2) ? "Yes" : "No";  
System.out.println(answer);
```

Terminologia

- Os principais termos usados nesta aula foram:
 - Instruções If
 - Instruções If-else
 - Scanner
 - instruções switch (case, switch e default)
 - Operadores Ternários

Resumo

- Nesta aula, você deverá ter aprendido a:
 - Usar Scanner para entrada do usuário durante a execução do programa
 - Usar lógica e instruções if-else
 - Aplicar lógica e instruções de switch no código Java
 - Usar quebra e padrão de forma efetiva em uma instrução de switch
 - Usar o operador ternário



The Oracle Academy logo is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font, with the letters "R" and "A" slightly overlapping. Below "ORACLE" is the word "Academy" in a dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

ORACLE

Academy