Aqui estão as soluções dos exercícios, com código em Java, explicações detalhadas e comentários destinados a alunos que estão a começar.

### **Exercício 1: Método "Olá Mundo!"**

**Código:**

// Esta é uma classe em Java chamada "OlaMundo"  
public class OlaMundo {  
 // Este é o método principal, o ponto de entrada do programa  
 public static void main(String[] args) {  
 // Aqui estamos a imprimir a mensagem "Olá Mundo!" na consola  
 System.out.println("Olá Mundo!");  
 }  
}

**Explicação:**

1. **Classe**: O Java organiza o código em classes. Aqui, criamos uma classe chamada OlaMundo.
2. **Método principal**: O método main é o ponto de partida de qualquer programa em Java. Quando o programa é executado, o método main é chamado automaticamente.
3. **Instrução de impressão**: O comando System.out.println exibe mensagens no ecrã.

### **Exercício 2a: Variável Inteira**

**Código:**

public class VariavelIdade {  
 public static void main(String[] args) {  
 // Declaração de uma variável inteira chamada "idade"  
 int idade;  
 // Atribuição do valor 21 à variável "idade"  
 idade = 21;  
 // Imprime a idade na consola  
 System.out.println("A idade é: " + idade);  
 }  
}

**Explicação:**

1. **Declaração**: Criámos uma variável chamada idade do tipo int (números inteiros).
2. **Atribuição**: Atribuímos o valor 21 a essa variável.
3. **Impressão**: Usámos System.out.println para exibir o valor na consola.

### **Exercício 2b: Constante Longa**

**Código:**

public class ConstanteCartao {  
 public static void main(String[] args) {  
 // Declaração de uma constante longa chamada "cartaoCidadao"  
 final long cartaoCidadao = 4110141171L;  
 // Imprime o valor da constante na consola  
 System.out.println("O número do Cartão de Cidadão é: " + cartaoCidadao);  
 }  
}

**Explicação:**

1. **Constante**: Usámos a palavra-chave final para criar uma constante, ou seja, um valor que não pode ser alterado.
2. **Tipo Longo**: O tipo long é usado para números maiores que os suportados por int. O L no final do número indica que é um long.

### **Exercício 3: Conversão de Float para Int**

**Código:**

public class ConversaoFloatParaInt {  
 public static void main(String[] args) {  
 // Declaração de um número do tipo float  
 float numeroFloat = 271.828f;  
 // Conversão do número float para int  
 int numeroInteiro = (int) numeroFloat;  
 // Imprime o número inteiro  
 System.out.println("O número convertido é: " + numeroInteiro);  
 }  
}

**Explicação:**

1. **Número Flutuante**: Declarámos uma variável float com o valor 271.828. O f no final indica que é um número float.
2. **Conversão**: Usámos (int) para converter o número float em int. A parte decimal é descartada.
3. **Impressão**: Mostrámos o valor convertido na consola.

### **Exercício 4: Manipulação de Strings**

**Código:**

public class ManipulacaoString {  
 public static void main(String[] args) {  
 // Declaração de uma string com o alfabeto  
 String txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
 // Demonstração de métodos úteis para strings  
 System.out.println("Comprimento da string: " + txt.length());  
 System.out.println("Primeira letra: " + txt.charAt(0));  
 System.out.println("Em minúsculas: " + txt.toLowerCase());  
 System.out.println("Subsequência de letras (C a F): " + txt.substring(2, 6));  
 System.out.println("Letra Z está na posição: " + txt.indexOf('Z'));  
 }  
}

**Explicação:**

1. **length**: Calcula o tamanho da string.
2. **charAt**: Mostra o carácter numa posição específica.
3. **toLowerCase**: Converte o texto para minúsculas.
4. **substring**: Extrai uma parte da string.
5. **indexOf**: Encontra a posição de uma letra específica.

### **Exercício 5: Resolução de Equação Quadrática**

**Código:**

import java.util.Scanner;  
  
public class EquacaoQuadratica {  
 public static void main(String[] args) {  
 // Criar um Scanner para ler os coeficientes  
 Scanner scanner = new Scanner(System.in);  
   
 // Pedir os coeficientes  
 System.out.print("Digite o valor de a: ");  
 double a = scanner.nextDouble();  
 System.out.print("Digite o valor de b: ");  
 double b = scanner.nextDouble();  
 System.out.print("Digite o valor de c: ");  
 double c = scanner.nextDouble();  
   
 // Calcular o discriminante  
 double delta = b \* b - 4 \* a \* c;  
   
 // Verificar se há soluções reais  
 if (delta >= 0) {  
 // Calcular as soluções  
 double x1 = (-b + Math.sqrt(delta)) / (2 \* a);  
 double x2 = (-b - Math.sqrt(delta)) / (2 \* a);  
 System.out.println("As soluções são:");  
 System.out.println("x1 = " + x1);  
 System.out.println("x2 = " + x2);  
 } else {  
 System.out.println("A equação não tem soluções reais.");  
 }  
   
 // Fechar o scanner  
 scanner.close();  
 }  
}

**Explicação:**

1. **Entrada de Dados**: Usámos a classe Scanner para ler os coeficientes a, b, c do utilizador.
2. **Cálculo do Discriminante**: A fórmula b² - 4ac é usada para verificar se a equação tem soluções reais.
3. **Fórmula Quadrática**: Calculámos x1 e x2 usando a fórmula (-b ± √delta) / 2a.
4. **Condição**: Verificámos se delta é negativo para informar que não há soluções reais.

Aqui estão as soluções detalhadas para os novos exercícios, com explicações pedagógicas e comentários no código.

### **Exercício 6: Jogo "Adivinha a Idade"**

**Código:**

import java.util.Scanner;  
  
public class AdivinhaIdade {  
 public static void main(String[] args) {  
 // Geração de uma idade aleatória entre 1 e 100  
 int idadeAleatoria = (int) (Math.random() \* 100) + 1;  
 Scanner scanner = new Scanner(System.in);  
   
 int palpite = 0;  
 System.out.println("Tente adivinhar a idade (entre 1 e 100)!");  
   
 // Ciclo para permitir múltiplas tentativas  
 while (palpite != idadeAleatoria) {  
 System.out.print("Qual é o seu palpite? ");  
 palpite = scanner.nextInt();  
   
 if (palpite < idadeAleatoria) {  
 System.out.println("Demasiado baixo! Tente outra vez.");  
 } else if (palpite > idadeAleatoria) {  
 System.out.println("Demasiado alto! Tente outra vez.");  
 } else {  
 System.out.println("Parabéns! A idade era " + idadeAleatoria);  
 }  
 }  
   
 scanner.close();  
 }  
}

**Explicação:**

1. **Geração de número aleatório**: Usámos Math.random() para gerar um número entre 0.0 e 1.0 e ajustámos para um intervalo entre 1 e 100.
2. **Leitura do palpite**: A classe Scanner foi usada para permitir que o utilizador insira valores.
3. **Lógica de feedback**: O programa informa se o palpite é maior ou menor do que o valor gerado até o utilizador acertar.

### **Exercício 7: Uso do Operador Ternário**

**Código:**

public class OperadorTernario {  
 public static void main(String[] args) {  
 // Declaração de uma idade  
 int idade = 18;  
 // Verificação se é maior de idade usando operador ternário  
 String status = (idade >= 18) ? "Maior de idade" : "Menor de idade";  
 // Imprime o resultado  
 System.out.println("A pessoa é: " + status);  
 }  
}

**Explicação:**

1. **Operador Ternário**: (condição) ? valorSeVerdade : valorSeFalso é uma forma curta de escrever um if-else.
2. **Exemplo**: Aqui verificámos se a idade é maior ou igual a 18 e atribuimos "Maior de idade" ou "Menor de idade" conforme o caso.

### **Exercício 8: Números Ímpares com While**

**Código:**

public class NumerosImpares {  
 public static void main(String[] args) {  
 int numero = 1;  
  
 // Ciclo para percorrer números de 1 a 15  
 while (numero <= 15) {  
 if (numero == 7) {  
 // Ignora o número 7  
 numero++;  
 continue;  
 }  
 if (numero % 2 != 0) {  
 // Imprime apenas os números ímpares  
 System.out.println(numero);  
 }  
 numero++;  
 }  
 }  
}

**Explicação:**

1. **Ciclo While**: Executámos um ciclo enquanto numero <= 15.
2. **Continue**: Saltámos o número 7 com a instrução continue.
3. **Números Ímpares**: Verificámos se o número é ímpar usando o operador %.

### **Exercício 9: Números Aleatórios com For**

**Código:**

public class GerarNumerosAleatorios {  
 public static void main(String[] args) {  
 // Gerar até 10 números aleatórios  
 for (int i = 0; i < 10; i++) {  
 double numero = Math.random();  
 System.out.println("Número gerado: " + numero);  
 // Interrompe o ciclo se o número for menor que 0.2  
 if (numero < 0.2) {  
 System.out.println("Número menor que 0.2 encontrado. Ciclo interrompido.");  
 break;  
 }  
 }  
 }  
}

**Explicação:**

1. **Ciclo For**: Executámos até 10 iterações.
2. **Math.random()**: Gerámos números aleatórios entre 0.0 e 1.0.
3. **Break**: O ciclo é interrompido com break se o número gerado for inferior a 0.2.

### **Exercício 10: Alfabeto Invertido**

**Código:**

public class AlfabetoInvertido {  
 public static void main(String[] args) {  
 // Declaração da string com o alfabeto  
 String txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
  
 // Percorrer a string em ordem inversa  
 for (int i = txt.length() - 1; i >= 0; i--) {  
 System.out.print(txt.charAt(i));  
 }  
 System.out.println(); // Para finalizar com uma nova linha  
 }  
}

**Explicação:**

1. **String com Alfabeto**: Declarámos a string txt com o alfabeto.
2. **Ciclo Inverso**: Percorremos a string do último carácter (txt.length() - 1) até o primeiro (i >= 0).
3. **Impressão**: Usámos charAt(i) para obter cada letra e System.out.print para exibir na mesma linha.

Aqui estão as soluções detalhadas para os novos exercícios, com explicações passo a passo e comentários no código.

### **Exercício 11: Idade do Universo**

**Código:**

public class IdadeDoUniverso {  
 public static void main(String[] args) {  
 // Declaração de uma variável do tipo long  
 long idadeDoUniverso;  
 // Atribuição do valor 13,8 mil milhões  
 idadeDoUniverso = 13\_800\_000\_000L;  
 // Impressão do valor na consola  
 System.out.println("A idade do universo é aproximadamente: " + idadeDoUniverso + " anos.");  
 }  
}

**Explicação:**

1. **Tipo long**: Usámos long para representar valores muito grandes, como 13,8 mil milhões.
2. **Separador \_**: Tornámos o número mais legível com \_, um recurso permitido no Java.
3. **Impressão**: Usámos System.out.println para exibir a idade.

### **Exercício 12: Constante e Variável short**

**Código:**

public class MemoriaShort {  
 public static void main(String[] args) {  
 // Declaração de uma constante do tipo short  
 final short memoria = 32767;  
 // Copiar o valor para uma variável do mesmo tipo  
 short memoriaVariavel = memoria;  
 // Imprimir o valor da variável  
 System.out.println("Valor inicial da variável: " + memoriaVariavel);  
 // Acrescentar uma unidade  
 memoriaVariavel++;  
 // Imprimir o valor após o incremento  
 System.out.println("Valor após incremento: " + memoriaVariavel);  
 // Explicação do resultado  
 System.out.println("O valor obtido deve-se ao overflow do tipo short, que retorna ao menor valor permitido (-32768).");  
 }  
}

**Explicação:**

1. **Constante short**: final define uma constante que não pode ser alterada.
2. **Cópia**: Atribuímos o valor da constante a uma variável.
3. **Overflow**: Quando incrementamos 32767, o valor ultrapassa o limite de um short e "volta ao início", resultando em -32768.

### **Exercício 13: Alternar Maiúsculas e Minúsculas**

**Código:**

public class AlternarLetras {  
 public static void main(String[] args) {  
 // Declaração da string com o alfabeto  
 String txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
 StringBuilder resultado = new StringBuilder();  
   
 // Percorrer todos os caracteres da string  
 for (int i = 0; i < txt.length(); i++) {  
 char letra = txt.charAt(i);  
 // Alternar maiúsculas e minúsculas  
 if (i % 2 == 1) {  
 resultado.append(Character.toLowerCase(letra)); // Minúsculas  
 } else {  
 resultado.append(letra); // Mantém maiúsculas  
 }  
 }  
   
 // Imprime o resultado  
 System.out.println("String resultante: " + resultado);  
 }  
}

**Explicação:**

1. **Percorrer a String**: Usámos um ciclo for para processar cada carácter.
2. **Minúsculas**: A função Character.toLowerCase() converte para minúsculas.
3. **StringBuilder**: Usámos StringBuilder para construir eficientemente a nova string.

### **Exercício 14: Exemplo de switch**

**Código:**

public class ExemploSwitch {  
 public static void main(String[] args) {  
 int dia = 3; // Representa terça-feira  
  
 // Exemplo de uso do switch para dias da semana  
 switch (dia) {  
 case 1:  
 System.out.println("Segunda-feira");  
 break;  
 case 2:  
 System.out.println("Terça-feira");  
 break;  
 case 3:  
 System.out.println("Quarta-feira");  
 break;  
 case 4:  
 System.out.println("Quinta-feira");  
 break;  
 case 5:  
 System.out.println("Sexta-feira");  
 break;  
 case 6:  
 case 7:  
 System.out.println("Fim de semana");  
 break;  
 default:  
 System.out.println("Dia inválido");  
 }  
 }  
}

**Explicação:**

1. **Switch**: Escolhe um bloco de código com base no valor de dia.
2. **Case múltiplo**: Agrupámos 6 e 7 para tratar ambos como fim de semana.
3. **Default**: O bloco default trata casos fora do esperado.

### **Objetivo do Exercício 15 - Explicação Detalhada**

Calcular a **média** de uma amostra de números fornecidos pelo utilizador. A média é um valor que representa o "centro" ou o "valor típico" da amostra.

### **Passo a Passo do Código**

import java.util.Scanner; // Importa a classe Scanner para ler dados do utilizador

**O que esta linha faz?**

* O Scanner é uma ferramenta que nos permite receber entradas do utilizador.
* Será usada para o utilizador fornecer os valores da amostra.

public class MediaAmostra {  
 public static void main(String[] args) {

**O que está a acontecer aqui?**

* Definimos uma classe chamada MediaAmostra.
* Dentro desta classe, temos o método main, que é o ponto de entrada do programa, ou seja, onde o programa começa a ser executado.

Scanner scanner = new Scanner(System.in);

**O que esta linha faz?**

* Criamos um objeto do tipo Scanner chamado scanner, que será usado para ler dados do utilizador.
* O System.in indica que os dados serão lidos a partir do teclado.

System.out.print("Digite o tamanho da amostra: ");  
int tamanho = scanner.nextInt();

**O que acontece aqui?**

1. O programa pede ao utilizador para introduzir o tamanho da amostra (quantos números ele vai fornecer).
2. Usamos o método nextInt() para ler um número inteiro (quantidade de valores na amostra) e armazená-lo na variável tamanho.

double soma = 0.0;

**Porquê esta linha?**

* Criamos uma variável chamada soma para guardar a soma de todos os valores que o utilizador vai fornecer.
* Inicializamos com 0.0 porque ainda não somámos nada.

for (int i = 1; i <= tamanho; i++) {  
 System.out.print("Digite o valor " + i + ": ");  
 double valor = scanner.nextDouble();  
 soma += valor; // Adicionar o valor à soma  
}

**Explicação do ciclo for:**

1. **Inicialização do ciclo**:
   * O ciclo começa com i = 1 (o primeiro número que o utilizador fornecerá).
2. **Condição**:
   * O ciclo continua enquanto i <= tamanho (garantimos que o utilizador forneça exatamente a quantidade de números indicada).
3. **Incremento**:
   * Em cada iteração, aumentamos i em 1 (i++), para pedir o próximo valor.

**O que acontece dentro do ciclo?**

* Pedimos ao utilizador para fornecer o valor -ésimo da amostra.
* Usamos scanner.nextDouble() para ler um número decimal e guardá-lo na variável valor.
* Adicionamos esse valor à variável soma com a operação soma += valor.

**Exemplo:**

* O utilizador introduz os valores 2.0, 4.0 e 6.0.
* A soma será calculada assim:
  + Inicialmente: soma = 0.0.
  + Após o primeiro valor: soma = 0.0 + 2.0 = 2.0.
  + Após o segundo valor: soma = 2.0 + 4.0 = 6.0.
  + Após o terceiro valor: soma = 6.0 + 6.0 = 12.0.

double media = soma / tamanho;

**O que esta linha faz?**

* Dividimos a soma de todos os valores (soma) pelo número de valores na amostra (tamanho) para obter a média.

**Exemplo:**

* Soma:
* Tamanho:
* Média:

System.out.println("A média dos valores da amostra é: " + media);

**O que acontece aqui?**

* Imprimimos o valor da média na consola, mostrando o resultado ao utilizador.

scanner.close();

**Porquê esta linha?**

* Fechamos o objeto scanner para liberar recursos do sistema.

### **Exemplo de Funcionamento**

**Entrada do Utilizador:**

* O utilizador introduz:
  + Tamanho da amostra: 3
  + Valores: 2.0, 4.0, 6.0

**O que acontece no programa?**

1. Soma:
2. Média:

**Saída do Programa:**

A média dos valores da amostra é: 4.0

### **Resumo do Fluxo do Programa**

1. Pedimos ao utilizador quantos números ele vai fornecer (tamanho da amostra).
2. Criamos uma variável soma para armazenar a soma dos valores.
3. Usamos um ciclo para pedir ao utilizador cada valor da amostra.
4. Somamos esses valores e calculamos a média dividindo pela quantidade de números.
5. Mostramos o resultado ao utilizador.

### **Exercício 16 - Explicação Simples e Detalhada**

O objetivo deste exercício é calcular a **média** e o **desvio padrão** de uma amostra de números fornecidos pelo utilizador. Vamos explicar passo a passo.

#### **Passo 1: O que é a Média?**

A média é a soma de todos os valores dividida pelo número de valores. Por exemplo:

* Se temos os números 2, 4, 6:
  + Soma =
  + Número de valores = 3
  + Média =

#### **Passo 2: O que é o Desvio Padrão?**

O desvio padrão mede o quanto os valores estão "espalhados" em relação à média. Se os valores estão muito próximos da média, o desvio padrão é pequeno. Se estão muito afastados, o desvio padrão é maior.

A fórmula do desvio padrão para uma amostra é:

* : Cada valor da amostra.
* : A média da amostra.
* : Número de valores na amostra.

**Passos para calcular o desvio padrão:**

1. Calcular a média ().
2. Para cada valor (), calcular a diferença para a média ().
3. Elevar essa diferença ao quadrado ().
4. Somar todos os quadrados.
5. Dividir essa soma pelo número de valores ().
6. Tirar a raiz quadrada do resultado.

#### **Código Explicado Passo a Passo**

Aqui está o código com explicação detalhada:

import java.util.Scanner;  
  
public class MediaDesvioPadrao {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 // 1. Pedir o número de valores na amostra  
 System.out.print("Digite o tamanho da amostra: ");  
 int tamanho = scanner.nextInt();  
  
 // 2. Criar um array para guardar os valores  
 double[] valores = new double[tamanho];  
 double soma = 0.0;  
  
 // 3. Ler os valores do utilizador  
 for (int i = 0; i < tamanho; i++) {  
 System.out.print("Digite o valor " + (i + 1) + ": ");  
 valores[i] = scanner.nextDouble(); // Guarda o valor no array  
 soma += valores[i]; // Soma o valor ao total  
 }  
  
 // 4. Calcular a média  
 double media = soma / tamanho;  
  
 // 5. Calcular a soma dos quadrados das diferenças  
 double somaDiferencas = 0.0;  
 for (double valor : valores) {  
 somaDiferencas += Math.pow(valor - media, 2); // (x\_i - média)^2  
 }  
  
 // 6. Calcular o desvio padrão  
 double desvioPadrao = Math.sqrt(somaDiferencas / tamanho);  
  
 // 7. Mostrar os resultados  
 System.out.println("Média: " + media);  
 System.out.println("Desvio Padrão: " + desvioPadrao);  
  
 scanner.close();  
 }  
}

#### **Passo a Passo Simplificado**

1. **Pedir o Tamanho da Amostra**:
   * O programa começa pedindo ao utilizador quantos números vai introduzir.
2. **Guardar os Valores e Somar**:
   * Criamos um **array** para guardar os números fornecidos pelo utilizador.
   * À medida que lemos os números, somamo-los para calcular a média depois.
3. **Calcular a Média**:
   * Dividimos a soma de todos os números pelo número total de valores.
4. **Somar as Diferenças ao Quadrado**:
   * Para cada número, calculamos a diferença entre ele e a média.
   * Elevamos essa diferença ao quadrado e somamos.
5. **Calcular o Desvio Padrão**:
   * Dividimos a soma das diferenças ao quadrado pelo número de valores.
   * Tiramos a raiz quadrada do resultado para obter o desvio padrão.
6. **Mostrar o Resultado**:
   * Imprimimos a média e o desvio padrão.

#### **Exemplo de Funcionamento**

* O utilizador fornece os números: 2, 4, 6.
* **Média**:
  + .
* **Desvio Padrão**:
  + , , .
  + Soma das diferenças ao quadrado = .
  + Dividido pelo número de valores = .
  + Raiz quadrada = .
* O programa mostra:
* Média: 4.0  
  Desvio Padrão: 1.63

#### **Conceitos Importantes a Reter:**

1. **Arrays**:
   * Usamos um array para guardar os valores da amostra. Um array é uma "tabela de números"
   * Isto permite reutilizar os valores para cálculos adicionais.
2. **Cálculo em Etapas**:
   * Primeiro calculamos a média, depois o desvio padrão. Cada cálculo depende do anterior.
3. **Uso de Math.pow e Math.sqrt**:
   * Estas funções ajudam a fazer cálculos matemáticos como potências e raízes quadradas.

### **Exercício 17: Número Primo**

**Código:**

import java.util.Scanner;  
  
public class NumeroPrimo {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 // Pedir um número ao utilizador  
 System.out.print("Digite um número: ");  
 int numero = scanner.nextInt();  
  
 // Verificar se o número é primo  
 boolean primo = true;  
  
 // Números menores ou iguais a 1 não são primos  
 if (numero <= 1) {  
 primo = false;  
 } else {  
 // Testar divisores de 2 até a raiz quadrada do número  
 for (int i = 2; i <= Math.sqrt(numero); i++) {  
 if (numero % i == 0) {  
 primo = false; // Encontrou um divisor, não é primo  
 break; // Sair do ciclo, já sabemos que não é primo  
 }  
 }  
 }  
  
 // Imprimir o resultado  
 if (primo) {  
 System.out.println("O número " + numero + " é primo.");  
 } else {  
 System.out.println("O número " + numero + " não é primo.");  
 }  
  
 scanner.close();  
 }  
}

**Explicação detalhada:**

1. **Entrada do utilizador**: Pedimos um número com Scanner. Este número será testado para verificar se é primo.
2. **Definição de primo**:
   * Um número primo é um número maior que 1, divisível apenas por 1 e por ele mesmo.
   * Exemplo: 5 é primo porque só pode ser dividido por 1 e 5. Já 4 não é primo porque é divisível por 2.
3. **Otimização com raiz quadrada**:
   * Para saber se um número é primo, não precisamos testar todos os números de 2 até n-1. É suficiente testar até a raiz quadrada de n.
   * Isto torna o programa mais eficiente.
4. **Lógica do ciclo**:
   * Se encontrarmos um divisor (numero % i == 0), podemos determinar que o número não é primo e interromper o ciclo.
5. **Saída**:
   * Informamos ao utilizador se o número é primo ou não.

### **Exercício 18: Números Ímpares Decrescentes**

**Código:**

public class NumerosImparesDecrescentes {  
 public static void main(String[] args) {  
 int numero = 30;  
  
 // Ciclo decrescente  
 while (numero >= 15) {  
 if (numero == 21) {  
 numero--; // Ignora o número 21  
 continue; // Passa para a próxima iteração  
 }  
 if (numero % 2 != 0) {  
 System.out.println(numero); // Imprime números ímpares  
 }  
 numero--;  
 }  
 }  
}

**Explicação detalhada:**

1. **Intervalo decrescente**:
   * Começamos com 30 e vamos até 15, diminuindo 1 a cada iteração (numero--).
2. **Verificar números ímpares**:
   * Um número é ímpar se numero % 2 != 0, ou seja, o resto da divisão por 2 não é zero.
3. **Ignorar o número 21**:
   * Usamos a instrução continue para saltar o processamento do número 21 e passar para a próxima iteração.
4. **Impressão**:
   * Apenas os números ímpares que não sejam 21 são impressos no ecrã.

### **Exercício 19: Separar Nome em Palavras**

**Código:**

import java.util.Scanner;  
  
public class NomeSeparado {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 // Pedir o nome completo ao utilizador  
 System.out.print("Digite o seu nome completo: ");  
 String nomeCompleto = scanner.nextLine();  
  
 // Separar as palavras do nome com base nos espaços  
 String[] palavras = nomeCompleto.split(" ");  
  
 // Imprimir cada palavra numa linha separada  
 System.out.println("O nome separado em palavras é:");  
 for (String palavra : palavras) {  
 System.out.println(palavra);  
 }  
  
 scanner.close();  
 }  
}

**Explicação detalhada:**

1. **Entrada do utilizador**:
   * Pedimos o nome completo com nextLine(), para incluir todo o texto digitado, incluindo espaços.
2. **Separação das palavras**:
   * Usamos o método split(" ") da classe String para dividir o texto em palavras.
   * O parâmetro " " indica que estamos a separar com base nos espaços.
   * O resultado é um array de strings, onde cada elemento corresponde a uma palavra do nome.
3. **Ciclo For**:
   * Percorremos o array e imprimimos cada palavra numa linha separada.
4. **Resultado**:
   * Por exemplo, para o nome "Luís Simões da Cunha", o programa imprime:
   * Luís  
     Simões  
     da  
     Cunha

### **Exercício 20: Palavra Mais Longa**

**Código:**

import java.util.Scanner;  
  
public class PalavraMaisLonga {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 // Pedir uma frase ao utilizador  
 System.out.print("Digite uma frase: ");  
 String frase = scanner.nextLine();  
  
 // Separar as palavras da frase  
 String[] palavras = frase.split(" ");  
  
 // Encontrar a palavra mais longa  
 String palavraMaisLonga = "";  
 for (String palavra : palavras) {  
 if (palavra.length() > palavraMaisLonga.length()) {  
 palavraMaisLonga = palavra;  
 }  
 }  
  
 // Imprimir a palavra mais longa e o seu comprimento  
 System.out.println("A palavra mais longa é \"" + palavraMaisLonga + "\" com " + palavraMaisLonga.length() + " caracteres.");  
  
 scanner.close();  
 }  
}

**Explicação detalhada:**

1. **Entrada do utilizador**:
   * Pedimos uma frase com nextLine().
   * Exemplo: "Este é um exercício interessante para aprender Java".
2. **Separação das palavras**:
   * Usamos split(" ") para dividir a frase em palavras, criando um array onde cada elemento é uma palavra.
3. **Encontrar a palavra mais longa**:
   * Inicializamos palavraMaisLonga como uma string vazia.
   * Percorremos o array de palavras e, para cada palavra, verificamos se o comprimento dela (length()) é maior que o da palavra atual mais longa.
   * Se for, atualizamos palavraMaisLonga.
4. **Impressão do resultado**:
   * Após percorrer todas as palavras, imprimimos a palavra mais longa e o número de caracteres.
   * No exemplo dado, o programa imprime:
   * A palavra mais longa é "interessante" com 12 caracteres.