3.5 A palavra-chave **new** cria um novo objeto da classe especificada à direita da palavra-chave. Por isso, **new** é fundamental no processo de criação de objetos em Java é utilizada para instanciar classes e criar instâncias de objetos com base nessas classes.

3.6 Um **construtor padrão** é um tipo que é usado para inicializar dados dos objetos de uma classe. Ele é chamado quando um novo objeto é criado usando a palavra-chave "new".
Se uma classe tiver apenas um construtor padrão e nenhuma inicialização das variáveis de instância for realizada dentro desse construtor, as variáveis de instância serão inicializadas com os valores padrão correspondentes aos seus tipos.

3.7 Variáveis de instância armazenam dados para instâncias individuais de uma classe e sendo declaradas dentro do escopo da classe, mas fora dos métodos, e é acessível a todos os métodos e atributos da instância.
O propósito de uma **variável de instância** é permitir que cada objeto de uma classe tenha seu próprio estado individual, armazenando dados específicos para cada objeto.

3.8 As **classes `System` e `String`** estão no pacote java.lang, que é importado implicitamente em todos os arquivos de código-fonte e são as classes pré-definidas. Isso significa que você pode usar as classes `System` e `String` em seu aplicativo Java sem precisar importá-las explicitamente.

3.9 A **classe Scanner** está contida no pacote java.util, para utilizar em um programa sem importá-la, é necessário fazer referência usando o nome completo (java.util.Scanner), incluindo o pacote, sempre que precisar utilizá-la.

3.10 Os **métodos get e set** para uma variável de instância com o objetivo de oferecer encapsulamento, validação e controle de acesso aos dados da classe. Gerando assim, consistência, segurança e flexibilidade e melhor gerenciamento.

Set: é responsável por modificar o valor de uma variável de instância, podendo receber um parâmetro com o novo valor para a variável realizando as validações e operações necessárias antes de realizar a atribuição.
Get: é responsável por retornar o valor atual de uma variável de instância, podendo não possuir parâmetros e apenas retornar o valor armazenado na variável, obtendo o valor da variável e evitando acesso direto, possibilitando a ações adicionais antes de retornar o valor.

3.11 public class GradeBook {
    private String courseName; // Nome do curso
    private String instrutorName; // Nome do instrutor

    // Construtor que inicializa o nome do curso e do instrutor
    public GradeBook(String courseName, **String instrutorName**) {
        this.courseName = courseName;
        this.instrutorName = instrutorName;
    }
    // Método para configurar o nome do instrutor
    public void setInstrutorName(String instrutorName) {

```
        this.instrutorName = instrutorName;
    }
    // Método para recuperar o nome do instrutor
    public String getInstrutorName() {
        return instrutorName;
    }
    // Exibe uma mensagem de boas-vindas, nome do curso e nome do instrutor
    public void displayMessage() {
        System.out.printf("Welcome to the GradeBook for\n%s!\n", getCourseName());
        System.out.println("This course is presented by: " + instrutorName);
    }
    // Método para configurar o nome do curso
    public void setCourseName(String courseName) {
        this.courseName = courseName;
    }
    // Método para recuperar o nome do curso
    public String getCourseName() {
        return courseName;
    }
}
```

3.12

```
import java.util.Scanner;

public class AccountTest {
    public static void main(String[] args) {
            Scanner teclado = new Scanner(System.in);

        Account account1 = new Account(50.00); // Cria o objeto Account 1
        Account account2 = new Account(-7.53); // Cria o objeto Account 2

        // Exibe o saldo inicial de cada objeto
        System.out.printf("account1 balance: $%.2f\n", account1.getBalance());
        System.out.printf("account2 balance: $%.2f\n\n", account2.getBalance());

        System.out.print("Enter debit amount for account1: ");
        double debitAmount = teclado.nextDouble();

        System.out.printf("\nsubtracting %.2f from account1 balance\n\n", debitAmount);
        if (account1.debit(debitAmount)) {
            System.out.println("Debit amount processed successfully");
        } else {
            System.out.println("Debit amount exceeded account balance");
        }

        // Exibe os saldos
        System.out.printf("account1 balance: $%.2f\n", account1.getBalance());
        System.out.printf("account2 balance: $%.2f\n\n", account2.getBalance());
```

```
        System.out.print("Enter debit amount for account2: ");
        debitAmount = teclado.nextDouble();
        System.out.printf("\nsubtracting %.2f from account2 balance\n\n", debitAmount);
        if (account2.debit(debitAmount)) {
            System.out.println("Debit amount processed successfully");
        } else {
            System.out.println("Debit amount exceeded account balance");
        }

        // Exibe os saldos
        System.out.printf("account1 balance: $%.2f\n", account1.getBalance());
        System.out.printf("account2 balance: $%.2f\n", account2.getBalance());

        input.close();
    }
}
```