Programação Orientada a Objetos e Estrutura de Dados

Equipe 5: Arthur Lins Belarmino, Lucas kenich Soares, Pedro Dimitry Zirianoff, Victor Batista Wang

Java: Estrutura Sequencial e Estruturas de Controle: Condicionais

A estrutura sequencial em Java representa o fluxo linear de execução, onde as instruções são executadas uma após a outra. Já as estruturas de controle condicionais (como if, else if, else e switch) permitem a execução de blocos de código diferentes dependendo de condições lógicas, permitindo tomadas de decisão no programa.

Estruturas de Controle: Laços, Métodos e Introdução a Vetores

Os laços (for, while, do-while) são usados para repetir trechos de código enquanto uma condição for verdadeira. Métodos são blocos de código reutilizáveis que podem receber parâmetros e retornar valores. Vetores (arrays) são estruturas que armazenam múltiplos valores do mesmo tipo em posições indexadas.

Introdução a Objetos e Classes; Atributos; Construtores

Java é uma linguagem orientada a objetos. Classes são moldes para criar objetos, que são instâncias dessas classes. Atributos representam as características dos objetos. Construtores são métodos especiais usados para inicializar objetos no momento da criação.

Métodos; Herança; Orientações sobre o Projeto Interdisciplinar

Métodos definem comportamentos das classes e podem ser invocados pelos objetos. A herança permite que uma classe herde atributos e métodos de outra, promovendo o reuso de código. Essa etapa também inclui orientações iniciais para o desenvolvimento do Projeto Interdisciplinar e da pesquisa associada.

Encapsulamento de Dados; Modificadores de Acesso; Classes Abstratas; Métodos Abstratos; Interface

O encapsulamento protege os dados internos da classe usando modificadores de acesso (private, public, protected). Classes abstratas não podem ser instanciadas e servem de base para outras classes. Métodos abstratos são definidos, mas não implementados na classe abstrata. Interfaces definem contratos que classes podem implementar.

Polimorfismo; Associação; Composição; Agregação

O polimorfismo permite que objetos de diferentes classes sejam tratados como objetos da mesma superclasse, com comportamentos específicos. Associação representa o relacionamento entre objetos. Composição é uma forma forte de associação (um objeto depende do outro para existir), enquanto a agregação é mais fraca (um objeto pode existir independentemente do outro).

Recursividade; Introdução à Análise de Algoritmo

A recursividade é uma técnica onde uma função chama a si mesma para resolver subproblemas. A análise de algoritmos estuda o desempenho de algoritmos em termos de tempo de execução e uso de memória, geralmente descrito com notações como Big-O.

Algoritmos de Ordenação e Busca

Algoritmos de ordenação, como Bubble Sort, Insertion Sort e Quick Sort, organizam os dados em uma sequência ordenada. Algoritmos de busca, como busca linear e busca binária, localizam elementos dentro de estruturas de dados.

Estrutura de Dados do Tipo Lista

Listas são coleções lineares de elementos. Elas podem ser implementadas como listas ligadas (cada elemento aponta para o próximo) ou listas dinâmicas (como ArrayList em Java), que podem crescer dinamicamente.

Estrutura de Dados do Tipo Pilha e Fila

Pilhas (LIFO – último a entrar, primeiro a sair) e filas (FIFO – primeiro a entrar, primeiro a sair) são estruturas lineares fundamentais. Pilhas são usadas, por exemplo, em chamadas de função recursivas, enquanto filas são comuns em sistemas de atendimento e processamento de tarefas.

Estrutura de Dados do Tipo Árvore

Árvores são estruturas hierárquicas compostas por nós, com um nó raiz e nós filhos. Cada nó pode ter vários filhos, mas apenas um pai. Árvores binárias, em especial, são aquelas em que cada nó tem no máximo dois filhos.

Estrutura de Dados do Tipo Árvore (continuação) e Projeto

Continuação do estudo sobre árvores, com foco em árvores binárias de busca (BST), que mantêm os elementos ordenados. Também inclui orientações sobre a entrega do Projeto Interdisciplinar, relacionando os conceitos estudados com sua aplicação prática.

Introdução a Grafos

Grafos são estruturas compostas por vértices (nós) e arestas (ligações). Podem ser direcionados ou não, ponderados ou não, e são usados para modelar relações complexas, como redes sociais, mapas e sistemas de transporte.

Paginas que foram utilizados codigos de Programação Orientada a objeto

Visualizar Comprovante

Categoria	Conceitos Aplicados
POO	Classes, Herança, Encapsulamento, Instâncias de objetos (ImageView, Button), Polimorfismo com Listener
Estrutura de Dados	Uso de String, Bundle (chave-valor), manipulação de dados via intent

AnexoPix

Categoria	Conceitos Utilizados
POO	Classe, Herança, Encapsulamento, Instâncias de Objetos (Intent, File, ImageView, etc.), Polimorfismo (sobrescrita), Modularização com métodos
Estrutura de Dados	Tipos primitivos (double, String), Objetos (Bitmap, File, Date, Intent), Bundle (chave-valor), SharedPreferences, controle de exceções

ApiClient

Categoria	Conceitos Utilizados
POO	Classe utilitária, Encapsulamento, Método estático, Modularização, Singleton, Uso de Interface (ApiService)
Estrutura de Dados	String, objetos (Retrofit, GsonConverterFactory), manipulação de dados em JSON, Composição de objetos

Comprovante

Categoria	Conceitos Utilizados
POO	Classe, Atributos privados, Encapsulamento, Construtor, Getters, Modelagem de objeto
Estrutura de Dados	Tipo String, Agrupamento de dados relacionados em objeto personalizado

ComprovanteAdpter

Categoria	Conceitos Utilizados
POO	Classe, Atributos privados, Encapsulamento, Construtor, Getters, Modelagem de objeto
Estrutura de Dados	Lista (List), Iteração por índice, Manipulação de arquivos como objetos (File)

Corrida

Categoria	Conceitos Utilizados
POO	Herança, Interface, Encapsulamento, Modularização, Comunicação entre Objetos com Intent, Polimorfismo
Estrutura de Dados	Lista (List<LatLng>), JSON (JSONObject, JSONArray), Manipulação de tempo (Calendar), Codificação de rota (Polyline)

DatabaseHelper

Categoria Conceitos Utilizados

POO	Herança (extends SQLiteOpenHelper), Encapsulamento (atributos privados via métodos públicos), Modularização (classe separada para acesso ao banco de dados)
Estrutura de Dados	Cursor (estrutura para percorrer registros do banco), ContentValues (estrutura de chave-valor para inserções e updates), Strings dinâmicas para queries
Banco de Dados (SQLite)	Criação e atualização de tabelas (onCreate, onUpgrade), uso de FOREIGN KEY, execução de comandos SQL, controle de versão do banco
Android	Utilização de Context, uso de Log para debug, interações com banco por SQLiteDatabase, boas práticas de persistência local

DatabaseHelper

Categoria Conceitos Utilizados

POO	**Encapsulamento** (uso de objetos Comprovante, ComprovanteAdapter), **Modularização** (responsabilidade separada para visualização e dados), **Intent** (comunicação entre telas)
Estrutura de Dados	**List (List<Comprovante>), Cursor (para percorrer resultados do banco de dados), Laços de repetição (do...while para preencher lista)**
Banco de Dados (SQLite)	Consulta com filtro (listarComprovantesPorUsuario(userId)), uso de colunas específicas com getColumnIndexOrThrow, uso de Cursor
Android	**SharedPreferences** (persistência de login), **RecyclerView/GridLayoutManager** (exibição de dados em grade), **startActivity/Intent** (navegação entre telas)

MainActivity

Categoria	Conceitos Utilizados
POO	Modularização (separação de métodos para inicialização e validação), Encapsulamento (uso de objetos User, ApiService), Callback para respostas
Android	Views (EditText, Button), Eventos de clique (setOnClickListener), Intents (navegação entre Activities), Toast (feedback ao usuário)
Validação	Validação de campos de texto (verificação de vazio, regex para email e telefone, tamanho mínimo de senha)
Rede / API REST	Retrofit para chamadas assíncronas, uso de callbacks (enqueue), tratamento de respostas HTTP e erros
Segurança	Hashing de senha antes do envio (via SecurityUtils.hashPassword)
Fluxo Assíncrono	Chamadas encadeadas de API (verificar email -> cadastrar usuário), tratamento de sucesso e falha de requisição

Menu

Categoria	Conceitos Utilizados
POO	Modularização (separação da lógica em métodos e classes), Encapsulamento (variáveis privadas)
Android	Views (Button, ImageView), Eventos de clique (setOnClickListener), Intents (navegação entre telas)

MudarPagamento

Categoria	Conceitos Utilizados
POO	Modularização (uso de métodos e eventos encapsulados dentro da Activity)
Android	Views (Button), Eventos de clique (setOnClickListener), Intents (navegação entre Activities)
Interface UI	Edge-to-edge (uso de EdgeToEdge.enable para layout com ajuste às bordas do sistema), Manipulação de WindowInsets para ajustar padding conforme as barras do sistema

Perfil

Categoria	Conceitos Utilizados
POO	Modularização (separação lógica em Activities e métodos de evento)
Android	Views (Button, ImageView), Eventos de clique (setOnClickListener), Intents (navegação entre Activities),

Procurar

Categoria	Conceitos Utilizados
POO	Organização do código em métodos com responsabilidades específicas (obterLocalizacaoAtual, setupAutoComplete, buscarRotaReal, etc.)
Android	Intents, Permissões (ACCESS_FINE_LOCATION), Eventos de clique, Fragments (SupportMapFragment, AutocompleteSupportFragment), Volley para requisições HTTP
Google Maps	GoogleMap, LatLng, CameraUpdateFactory, PolylineOptions, Markers, API de Directions (rota entre origem e destino)
Interface UI	Autocomplete de endereços (Google Places), Toasts para feedback do usuário, ajustes visuais com zoom e centralização do mapa
API Web	Requisições HTTP com Volley para consumo da Google Directions API, manipulação de JSON (JSONObject, JSONArray)

TelaCorrida

Categoria	Conceitos Utilizados
POO	Encapsulamento da lógica da tela em uma Activity, uso de atributos e métodos para modularização
Android	Intent para navegação entre telas, getIntent().getStringExtra para passagem de dados entre Activities
Interface UI	TextView para exibição dinâmica de dados, botões com setOnClickListener para ações do usuário

TelaInicial

Categoria	Conceitos Utilizados
POO	Criação de objetos (LoginRequest, LoginResponse), organização da lógica dentro de métodos
Android	SharedPreferences para persistência de dados locais, Intent para navegação entre telas
UI Interativa	EditText, Button, TextView, Toast para interação e feedback do usuário
Segurança	Criptografia de senha com SecurityUtils.hashPassword() antes de enviar ao servidor
Validação	Verificação de campos vazios (isEmpty) antes de fazer o login
Logging	Log.e() para registrar erros da API em tempo de execução