Divisão de Tarefas - Projeto "Grafos: Produção & Logística Inteligente"

Objetivo Geral
Desenvolver um sistema em linguagem C que simula a cadeia de produção e logística utilizando grafos,

Equipe e Responsabilidades

Pessoa 1 - Produção & Confiança
Foco nos grafos de dependências (DAG) e confiança de fornecedores.

Responsabilidades:
- Criar e manipular grafos direcionados acíclicos para dependências de produção.
- Implementar ordenação topológica.
- Criar módulo de fornecedores com índice de confiança.
- Integrar confiabilidade como critério para escolha de fornecedores.

Arquivos principais:
- grafo.h / grafo.c
- producao.h / producao.c
- confianca.h / confianca.c
- tests/test_producao.c, test_confianca.c

Pessoa 2 - Logística & Interface
Foco nos grafos ponderados de rotas logísticas e interação via terminal.

Responsabilidades:
- Implementar grafo ponderado representando as rotas.
- Desenvolver algoritmos de menor caminho: Dijkstra e A*.
- Criar menus no terminal para manipulação do sistema.
- Simular falhas em rotas e testar cenários diversos.

Arquivos principais:
- logistica.h / logistica.c
- interface.h / interface.c
- utils.h
- tests/test_logistica.c, test_interface.c

Estrutura de Diretórios
/projeto_grafos/
■■■ src/
■ ■■■ main.c
■ ■■■ grafo.h / grafo.c
■ ■■■ producao.h / producao.c
■ ■■■ logistica.h / logistica.c
■ ■■■ confianca.h / confianca.c
■ ■■■ interface.h / interface.c
■ ■■■ utils.h
■■■ tests/
■ ■■■ test_producao.c
■ ■■■ test_logistica.c
■ ■■■ test_confianca.c
■ ■■■ test_interface.c
■■■ docs/
■ ■■■ relatorio.pdf

■■■ Makefile
■■■ README.md

Esqueleto Inicial Compartilhado

Estrutura de Grafo Base (grafo.h)

```
typedef struct Aresta {
    int destino;
    int peso;
    struct Aresta* prox;
} Aresta;

typedef struct Vertice {
    int id;
    Aresta* listaArestas;
} Vertice;

typedef struct Grafo {
    int numVertices;
    Vertice* vertices;
    int** matrizAdjacencia; // opcional
} Grafo;
```

Funções:

```
Grafo* criarGrafo(int numVertices);
void adicionarAresta(Grafo* grafo, int origem, int destino, int peso);
void imprimirGrafo(Grafo* grafo);
void liberarGrafo(Grafo* grafo);
```

Ordenação Topológica (producao.c)

```
void ordenacaoTopologica(Grafo* grafo, int* resultado);
```

Algoritmo de Dijkstra (logistica.c)

```
int* dijkstra(Grafo* grafo, int origem);
```

Algoritmo A* (logistica.c)

```
int* a_star(Grafo* grafo, int origem, int destino, int (*heuristica)(int, int));
```

Avaliação de Confiança (confianca.c)

```
typedef struct Fornecedor {
    int id;
    float confianca; // 0.0 a 1.0
} Fornecedor;

float getConfiancaFornecedor(int id);
```

Menu de Interface (interface.c)

```
void mostrarMenu();
void executarSimulacao(Grafo* grafoProducao, Grafo* grafoLogistica);
```

Convenções:

- Prefixo de funções por módulo (producao_, logistica_, conf_, menu_)
- Todas estruturas principais devem estar em grafo.h
- Usar nomes específicos como produto, rota, fornecedor etc.

Sugestão de Integração (main.c)

```
#include "grafo.h"
#include "producao.h"
#include "logistica.h"
#include "interface.h"

int main() {
    Grafo* grafoProducao = criarGrafo(10);
    Grafo* grafoLogistica = criarGrafo(10);

    mostrarMenu();
    executarSimulacao(grafoProducao, grafoLogistica);

    liberarGrafo(grafoProducao);
    liberarGrafo(grafoLogistica);
    return 0;
}
```