INSTITUTO FEDERAL
Minas Gerais

IFMG - Campus São João Evangelista
Sistemas de Informação
Turma: SI 221 2022-2

Professor: Eduardo Trindade
eduardo.trindade@ifmg.edu.br
Algoritmos e Estrutura de Dados I

Trabalho Prático II - 10 pontos

Observações

- O trabalho é **individual**.
- É recomendado discutir os problemas e estratégias de solução com seus colegas.
- Tente solucionar os problemas vocês mesmos, pois solucionar problemas e desenvolver o raciocínio lógico é componente fundamental neste curso.
- Dúvidas devem ser postadas no fórum para discussão (Trabalho Prático I) no Moodle. Suas dúvidas podem ser as mesmas de outros colegas. Evite acionar o professor fora de aula para isso!
- Serão sorteados de 1 a 3 alunos para apresentar seu trabalho para a turma.
- Toda e qualquer mensagem de orientação e de erro deve ser adequadamente tratada.
- Clareza, indentação e comentários no programa serão bem avaliados.
- Trabalhos identificados como cópias ou plágio serão punidos **a rigor**.
- Evite o uso de funções ou estruturas de dados que não tenha conhecimento para explicar. Códigos complexos ou que exigirem interpretação de dados, sem a devida documentação, não serão avaliados. É possível desenvolver o trabalho somente com os conteúdos reforçados em aula.
- A entrega deverá ser feita **exclusivamente** pelo Moodle.
- Em hipótese alguma serão aceitos trabalhos entregues fora do prazo informado no Moodle.
- **Atenção:** Entregar em um único arquivo .zip, (ou .rar ou .7z ou .tar.gz) contendo o(s) arquivo(s) necessário(s) para a compilação de seu projeto (exemplo: main.cpp). O nome do arquivo .zip **deve ser** TP2AEDS1SeuNome.zip.

Objetivos

- Fixar conceitos sobre estruturas de dados e modularização.
- Trabalhar com TAD Pilha Simples com Arranjo.
- Estimular o raciocínio lógico e a proposição para solução de problemas.

Descrição

Faça em C/C++, um sistema para o setor de *delivery* de uma Lanchonete. O sistema deve possuir um menu principal através do qual o funcionário da lanchonete pode escolher a opção que deseja. As operações (opções do menu principal) são:

1.) Inclusão de um novo pedido: Neste caso, deve-se entrar com o(s) código(s) dos produtos referentes ao pedido. Implemente uma forma para que o código do pedido seja gerado automaticamente. Na estrutura do pedido, crie uma variável do tipo *float* chamada *distancia*. Preencha um valor qualquer no momento de registro do pedido, correspondendo à distância entre o cliente e a lanchonete.

2.) Listar Pedidos: Liste todos os pedidos gerados com seus respectivos produtos e preço final.

3.) Ver Cardápio: Imprima nesta opção todos os produtos da lanchonete disponíveis para serem incluídos no pedido (não é necessário cadastro de novos produtos no sistema). Estes produtos podem ser pré-definidos em uma matriz ou vetor, já estando disponíveis para consulta no início do programa.

4.) Consultar Pedido: Mostra os dados de um pedido através da busca pelo seu código.

5.) Imprimir Lista de Entrega: Considere uma Estrutura de Dados Pilha para a criação da Lista de Entrega. Organize a TPilha *Mochila* para que os pedidos sejam inseridos por ordem decrescente da *distancia* do pedido, ou seja, no topo ficarão os pedidos mais próximos da lanchonete. Mostre a Lista de Entrega.

6.) Lançar Entrega: Remova do topo da Mochila o pedido entregue.

Implementação

A implementação deve seguir as seguintes regras:

(1) Todos os dados referentes aos pedidos devem ser armazenados em *structs* simples para facilitar o acesso. Um pedido deve conter pelo menos o código do pedido, um vetor com códigos de produtos lançados, uma variável com o valor total do pedido e uma variável para a distância do pedido até a lanchonete (m ou km).

(2) O sistema deverá trabalhar com um arquivo binário de registros de funcionário para armazenar os dados gerados pelo sistema de forma permanente. O arquivo deve ser denominado "lanchonete.bin". Ao se iniciar a execução do sistema, todos os registros do arquivo devem ser lidos e inseridos adequadamente; antes de se terminar a execução, deve ser feito o contrário: todos os registros devem ser gravados no arquivo. Desta forma, os dados gerados pelo sistema não serão perdidos.

(3) O sistema deve permitir a manipulação das *structs* possibilitando as inclusões, consultas e listagem de maneira eficiente.

(4) Poderão ser utilizadas todas as TAD vistas que auxiliarem na solução do problema.

(5) Poderão ser utilizadas as estruturas criadas em aula com as funcionalidades vistas, adaptadas para trabalhar com os dados de funcionários da empresa e projetos, respectivamente, bem como qualquer outra TAD criada ou reproduzida, desde que devidamente referenciada.

Entrega

O que deverá ser entregue:

1. **Código fonte bem indentado e comentado**: Código fonte contendo todas as especificações da etapa de Implementação e estruturas/funções bem definidas e tratadas.
2. **Documentação do Trabalho**: Relatório do trabalho em formato PDF seguindo o modelo fornecido (disponível no Moodle).
3. O sistema deve permitir a manipulação das *structs* possibilitando as inclusões, consultas e listagem de maneira eficiente.
4. Poderão ser utilizadas todas as TAD vistas que auxiliarem na solução do problema.

Avaliação

1. Código bem indentado, comentado e funcional: 3 pts
2. Uso correto de variáveis, funções, estruturas de dados, arquivos e alocação dinâmica: 1 pt
3. Utilização de TADs e funções bem definidas sem repetição de código: 1 pt
4. Modularização: 0,5 pts
5. Código disponível em repositório do Github: 0,5 pts
6. Documentação utilizando modelo: 4 pts (Introdução 1 pt, Desenvolvimento 2 pts, Conclusão 0,5 pts, Referências e Apêndices 0,5 pts)

Sugestão de Estrutura Básica

```c
#define MAXTAM 10
#define MAX_ENTREGA 7

typedef struct {
    int codigo;
    int produtos[MAXTAM];
    float valor_pedido;
    float distancia;
} Pedido;

typedef struct {
    Pedido Pilha[MAX_ENTREGA];
    int Topo;
} TPilha;

void TPilha_Inicializa(TPilha *p);

int TPilha_Vazia(TPilha *p);

int TPilha_Cheia(TPilha *p);

void TPilha_Empilha(TPilha *p, Pedido x);

int TPilha_Desempilha(TPilha *p);

void TPilha_Imprime(TPilha *p);

int TPilha_Tamanho(TPilha *p);
```