



UNIOESTE

Universidade Estadual
do Oeste do Paraná

3º Trabalho de Algoritmos e Estruturas de dados

Professor:

Rômulo César Silva

Grupo:

Marco A. Guerra Pedroso,
Milena Lucas Dos Santos,
Victor Emanuel Almeida

18 de julho de 2021

Conteúdo

1	Instruções para execução do programa	2
1.1	Compilando o programa	2
1.2	Arquivos para compilação e execução	2
1.3	Caracteres detectados pelo Menu	2
2	Estruturas de Dados	3
2.1	Estrutura do Menu	3
2.2	Estruturas do arquivo de índices	4
2.3	Estruturas do arquivo de Dados	4
2.4	Estruturas em memória	4

Lista de Figuras

1 Instruções para execução do programa

1.1 Compilando o programa

Todos os arquivos de implementação estão na pasta “src” e subdivididos nas pastas:

- controllers;
- interfaces;
- models;
- menu;

Sendo assim para realizar o processo de compilação em um sistema operacional que possui o compilador **GCC**, basta utilizar o comando “**gcc ../src/controllers/*.c ../src/interfaces/*.c ../src/models/*.c ../src/menu/*.c ../src/main.c**”

1.2 Arquivos para compilação e execução

Dentro da raiz do projeto existe uma pasta chamada build onde se encontram:

- Um arquivo de entrada;
- O script de compilação do programa (funciona apenas no Linux);
- Após a primeira execução 2 arquivos de dados:
 - index.bin: armazena os índices dos elementos da árvore;
 - data.bin: armazena todos os dados de cada produto;

É importante que uma vez gerado o executável o mesmo seja colocado dentro da pasta build, facilitando o caminho para o arquivo de entrada e deixando os binários gerados separados do código fonte.

1.3 Caracteres detectados pelo Menu

Durante a execução do menu, o mesmo aceita os seguintes entradas de dados:

- **ENTER**: o caractere faz com que execute a função escolhida;

-
- **ESC**: o caractere faz com que volte ao menu anterior ou encerra o programa caso esteja no menu principal;
 - **UP**: o caractere ‘**W**’ faz com que o item selecionado receba seu anterior, dentro de uma lista encadeada circular, ou seja o anterior do primeiro é o último;
 - **DOWN**: o caractere ‘**S**’ faz com que o item selecionado receba seu próximo, dentro de uma lista encadeada circular, ou seja o próximo do último é o primeiro;
 - **1–9 (dígitos)**: Quando a entrada é um dígito, o item selecionado se torna aquele com o número entrado, e quando essa entrada for igual a opção selecionada é equivalente a tecla “ENTER” supracitada. Por exemplo caso o usuário aperte “44”, o programa executará a função “Carregar lista de produtos.”.

2 Estruturas de Dados

Escrever algo aqui

2.1 Estrutura do Menu

```
typedef int CallbackFunc(ArgStack head);
typedef void HeaderFunc();
typedef void FooterFunc();

typedef struct entryNode {
    int number;
    char entryMessage[MESSAGE_SIZE];
    CallbackFunc *funct;
    struct entryNode *next;
    struct entryNode *prev;
}Entry;
typedef Entry* EntryList;

typedef struct {
    FooterFunc *footer;
    HeaderFunc *header;
    EntryList first;
    EntryList selected;
}Menu;
```

2.2 Estruturas do arquivo de índices

Dados gravados no arquivo “index.bin”

```
typedef struct {
    int regRoot;
    int regLast;
    int regFree;
}IndexHead;

typedef struct {
    int key;
    int position;
    int leftChild;
    int rightChild;
}RegistryField;
```

2.3 Estruturas do arquivo de Dados

Dados gravados no arquivo “data.bin”

```
typedef struct {
    int regLast;
    int regFree;
}DataHead;

typedef struct {
    int numberOfKeys;
    int key[ORDER];
    int position[ORDER];
    int children[ORDER + 1];
}Registry;
```

2.4 Estruturas em memória

```
#define MAX_NAME 51

#define MAX_LOCAL 101

typedef struct {
    int code;
```

```
char name[MAX_NAME];  
int number;  
float value;  
char local[MAX_LOCAL];  
}Product;
```
