Diagrama

Descrição gerada automaticamente com confiança baixa

**Faculdade de Ciências e Tecnologia**

**Licenciatura em Engenharia**

**Eletrotécnica e de Computadores**

**Estruturas de Dados e Algoritmos**

**Mini Projeto I**

**Jogo da Forca**

Trabalho realizador por:

Eduardo Rocha Falvo

(2021192252)

Gabriel Santos Corrêa da Silva

(2021104069)

PL6

19 de maio de 2022

Sumário

**1 Introdução ........................................................................................... 3**

**3 Estrutura Geral Parte 2 ....................................................................... 3**

3.1 Linked Lists ................................................................................................... 3

3.1.1 listaTemas ................................................................................................ 3

3.1.1 listaPalavras\_Tema ................................................................................. 4

3.2 Novos Modos ................................................................................................. 4

3.2.1 Avançado ................................................................................................. 4

3.2.1 Profissional .............................................................................................. 4

**4 Conclusão ............................................................................................ 7**

**1. Introdução**

Proposto como continuação do MiniProjeto I da cadeira de Estruturas de Dados e Algoritmos, neste relatório iremos dissertar à respeito do desenvolvimento do nosso MiniProjeto 2 que, utilizando a linguagem de programação C++ e suas bibliotecas disponíveis, criássemos nosso próprio jogo a partir de especificações predefinidas pelos professores da cadeira. Iremos abordar as metodologias e lógicas utilizadas pelos integrantes do grupo ao desenvolver e executar o algoritmo do jogo.

**3. Estrutura Geral Parte 2**

Ao analisar a lógica do Jogo da Forca e verificar as novas especificações e funcionalidades solicitadas pelos Professores; concluímos que a melhor estrutura a ser usada no desenvolvimento do programa adicional ao que já tínhamos seria a implementação da lógica de novas Linked Lists para manipulação dos dados em ficheiros binários.

Iniciamos a implementação do MiniProjeto 2 por separar nosso ficheiro principal em vários ficheiros ‘headers’ e cpp. Além da divisão, como já comentado no primeiro relatório do MiniProjeto 1, a partir de pesquisas a respeito da linguagem utilizada no projeto, nós descobrimos o fato de que, quando feito um “rewrite” de uma linha de um ficheiro binário em C++ todos os dados abaixo daquela linha são perdidos, de modo a nos fazer chegar a conclusão de que o mais adequado nos casos de alterações de dados nestes ficheiros seria importar toda a ‘data’ dele para uma linked list, fazer as alterações necessárias enquanto os dados estão neste espaço e só então reescrever o ficheiro binário do início (lógica utilizada em praticamente todas as novas implementações solicitadas).

Assim, descrito brevemente a linha de raciocínio do grupo ao enfrentar a resolução do projeto, vamos agora comentar e detalhar novas classes e atributos criados para a conclusão da tarefa.

**3.1 Linked Lists**

Adicionalmente à Linked List de Rankings existente no MiniProjeto 1, nesta versão, incluímos ainda outras duas Linked Lists, a listaTemas e a listaPalavras\_Tema que serão comentadas individualmente a seguir.

3.1.1 listaTemas

Sendo a linked list associada à classe Temas, que conta com o atributo ‘tema’ (estrutura com o nome do tema e a quantidade de palavras associadas a este tema), e responsável por manipular os temas do nosso ficheiro “Temas.dat”, listaTemas conta com alguns principais métodos. Primeiramente, criamos seu construtor e destrutor para criar e deletar a lista; além disso, criamos para inserir itens, removê-los e escrever a lista - todos os métodos padrões de linked lists geral.

Em segundo lugar, criamos o método individual da nossa linked list, que é usado principalmente no novo modo “Avançado” do nosso código ao permitir criar novos temas e adicioná-los a nossa lista de temas em “Temas.dat” já existentes, e que é o escreveFicheiro.

3.1.1 listaPalavras\_Tema

Sendo a linked list associada à classe ‘palavras\_Tema’, que conta com os atributos ‘nome\_C’ (char com a palavra) e ‘jaJogada’ (int com o número de vezes que ela foi jogada), e responsável por manipular as palavras de um ficheiro de tema específico, listaPalavras\_Tema conta com alguns principais métodos. Primeiramente, criamos seu construtor e destrutor para criar e deletar a lista; além disso, criamos um método para removê-los passando seu índice e escrever a lista - todos os métodos padrões de linked lists geral.

Em segundo lugar, criamos o método individual da nossa linked list, que é usado tanto na lógica do modo “Avançado” quanto “Profissional” do nosso código (ambos modos novos) ao permitir editar temas já existentes (adicionar nova palavra, reescrever uma palavra ou excluir uma palavra) que são os métodos:

* insereItem(), agora com dois parâmetros, um sendo a nova palavra para adicioná-la ao tema e outra sendo o número de jogadas que ela já teve (atributo de ‘palavras\_Tema’ que é ajuda a selecionar as palavras menos jogadas no modo ‘Profissional’).
* editaLista(), que ao receber o index e a palavra, pode adicionar, reescrever ou excluir uma palavra, como já dito.

**3.2 Novos Modos**

3.2.1 Avançado

Para o modo Avançado, começamos inicialmente por adicionar no menu de modos um item para ele. Após isso, criamos na classo principal ‘Jogo’ três métodos principais:

* adicionaTema(), método que conversa com listaTemas() e utilizada das estruturas de tema ‘STema’ (já existente) e palavra ‘SPalavras’ (criada para associar a uma palavra o número de jogadas que já teve – e que agora é o componente que guardamos em nosso ficheiro de tema) e abre o ficheiro binário e inclui o tema desejado, com ‘STema’, e, associado a esse tema, um mínimo de oito palavras a partir de um loop, com ‘SPalavras’.
* editaTema(), método que conversa com listaPalavras\_Tema() e permite, a partir da manipulação de linked lists comentada, adicionar nova palavra, reescrever uma palavra já existente ou excluí-la.
* excluitTema(), método que conversa com listaTemas() e utilizada das estruturas de tema ‘STema’ (já existente) para excluir um Tema específico da lista e reescrevê-la sem este tema. Além disso, conta com a função remove() passando como parâmetro o nome de um ficheiro binário para excluí-lo também e não ocupar memória desnecessária.

E por fim, além de podermos acessar o modo Avançado no menu de modos, podemos também fazer toda esta manipulação de temas e palavras na nova opção criada no menu principal, sem necessariamente iniciar um jogo após isto.

3.2.1 Profissional