**Relatório**

**Projeto 1 – Programação 3 – Anagrama**

1. **PermutacaoComRepeticoes** O método faz o que é suposto, printar todas as permutações possível por cada char afixada da string original

Nota: Deixei comentado os passos caso quisesse colocar tudo numa lista e depois retornar o mesmo, optei por não e explorar uma nova ideia que tive…

**Raciocínio**: Optei por fazer o print na consola para aprender mais como posso fazer operações que trabalhem com a força bruta quer na pesquisa, ordenação e qualquer outra operação.  
Claro que quando ele faz o print poderia eu ter acesso a este dado e fazer n operações ou validações de acordo com as minhas necessidades ou vontades. Por exemplo, encontre a primeira ocorrência na recursividade e faça algo…

**Explicando**: Este código está implementando uma função recursiva para encontrar todas as permutações de uma determinada string s com repetição de caracteres permitida. O parâmetro de resposta é usado para armazenar as permutações conforme elas são construídas.  
  
A função primeiro verifica se a string de entrada s está vazia. Se for, imprime a permutação atual armazenada no parâmetro de resposta e incrementa um contador chamado Result.  
  
Se s não estiver vazio, a função itera por cada caractere em s. Para cada caractere, ele remove esse caractere da string e o anexa à string de resposta. Em seguida, ele chama a si mesmo com os s modificados e as strings de resposta como argumentos. Esse processo continua até que s esteja vazio, ponto em que a permutação atual é impressa e o contador é incrementado.  
  
Observe que esta função não retorna uma lista de permutações. Em vez disso, ele imprime cada permutação à medida que é encontrada e acompanha o número total de permutações encontradas no contador Result.

1. **PermutacaoSemRepeticoes**

**Raciocínio**: Este método já trabalhei de outra forma, sempre explorando novas forma de entender e até mesmo de explicar, poderia fazer como no anterior, mas não, neste já tentei trabalhar com as listas e a sua manipulação.  
Por conveniência já deixei que o método fizesse o print na consola diretamente, mas poderia perfeitamente retornar uma lista e a medida que ia percorrendo o mesmo, ir printando os valores.

**Explicando**: Este código é uma função recursiva que gera todas as permutações de uma string s sem repetições. Ele faz isso usando uma abordagem de recursividade.  
  
A função tem três parâmetros:  
  
s: a string de entrada  
resposta: a permutação atual sendo construída  
lista: uma lista de permutações que já foram geradas, usada para evitar a geração de duplicatas  
O caso base da recursão é quando o comprimento de s é zero, caso em que a resposta da permutação atual é impressa no console e adicionada à lista de permutações lista.  
  
Caso contrário, a função itera por cada caractere ch em s, remove-o da string e o anexa à resposta de permutação atual. Em seguida, ele chama a si mesmo com os s e a resposta modificados e a lista original. Este processo continua até que o caso base seja alcançado.

1. **PermutacoesEmDicionario**

**Raciocínio:** Primeiro para testar o meu raciocínio fiz um ciclo for encadeado um dentro do outro, que me daria elementos de uma lista que estejam contidas na outra lista, mas após fazer o “Stopwatch” o tempo de execução era elevado, o que é obvio, pois é na ordem 0(N), então para otimizar o código e o programa como um todo, usei as estruturas de pesquisas binarias abordadas nas aulas de arvoreAVL que otimiza e muito a busca na base do seu Hashcode que eu vou adicionando a medida que vou **CarregarDicionario()**

**Explicando:** Método que tenta encontrar as permutações de uma lista de strings em um dicionário.  
O método recebe uma lista de strings como entrada e retorna uma lista de strings. Ele cria uma lista vazia chamada listaNova, que será utilizada para armazenar as permutações encontradas no dicionário.  
Em seguida, itera através de cada string na correspondência da lista de entrada. Para cada string, ele procura o código hash da string no dicionário arvore. Se o código hash for encontrado, o valor correspondente é recuperado e adicionado à lista listaNova. Por fim, o método retorna a lista listaNova.

Os restantes dos métodos achei por bem não colocar no relatório por serem pequenos demais ou por estarem bem explicadas nos comentários.

**Análise final**

O trabalho foi desafiador, pois no início não sabia muito bem por onde começar, mas acabei por me orientar e aprender muito com a ajuda dos fóruns da web.  
Consegui o meu objetivo que era completar pelo menos ¾ do projeto proposto pelo docente e acredito que consegui apresentar um trabalho que reflete o que temos aprendido ao longo do semestre na disciplina de programação 3.  
  
Guia de evolução do trabalho – encontras as data e o meu processo diário

GitHub ( <https://github.com/LeandroSantos3/Anagrama.git> )

**Referências de pesquisas**

<https://www.geeksforgeeks.org/c-program-to-print-all-permutations-of-a-given-string-2/>

<https://www.youtube.com/watch?v=GuTPwotSdYw&t=737s&ab_channel=TECHDOSE>

<https://www.youtube.com/watch?v=e5AWVvDxLNM&ab_channel=MadhaviParasaniya>

<https://www.youtube.com/watch?v=hQ8tge-f7NE&ab_channel=CodePlay>

<https://social.msdn.microsoft.com/Forums/pt-BR/3ba1adbb-0093-4ba4-b92e-cc23e75bb44c/algoritmo-para-obter-combinaes?forum=clientept>