**Relatório do Trabalho de Implementação**

A base de dados Original, a compartilhada com toda turma, possuía os textos separados por aspas duplas, por isso dificultar na hora da busca pelas strings, resolvi substituir as aspas duplas por aspas simples e gerei um outro arquivo txt, que consta neste link:

<https://drive.google.com/file/d/1qzvYOpnsd574EIaR3wlyl6uEatVgJ3hB/view?usp=sharing>

1. **Leitura e geração de base de dados somente de Tweets.**

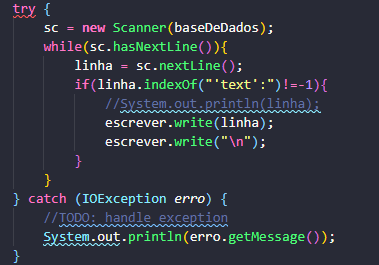
Eu separei o projeto em 2 projetos diferentes, um primeiro lê essa base de dados e gera um arquivo txt que contém apenas os Tweets que precisarei para o trabalho, essa parte de geração foi feita no projeto chamado “Gerador de Tweets”, nele há uma classe chamada “Lendo” na qual eu gero o txt só com os Tweets que utilizarei no outro projeto:



Em “baseDeDados” eu coloco o caminho do arquivo que consta com minha base de dados separada por aspas simples.

“tweets” eu criei um arquivo no qual ficará armazenado somente as linhas que contem o campo “text:” que é onde fica localizado os Tweets.

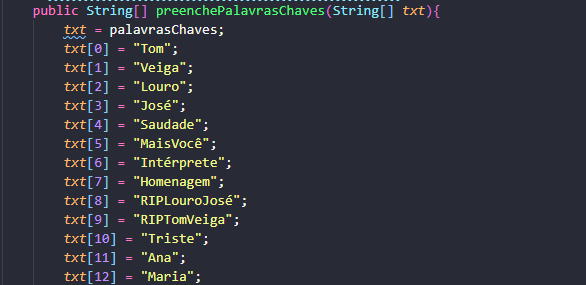
Na próxima imagem eu mostrarei a principal função dessa classe que é a parte da busca dos Tweets no campo “Text:”, quando o retorno dá diferente de -1 significa que há um tweet e eu escrevo no meu arquivo de saída “Tweets.txt”, como mostra a imagem:



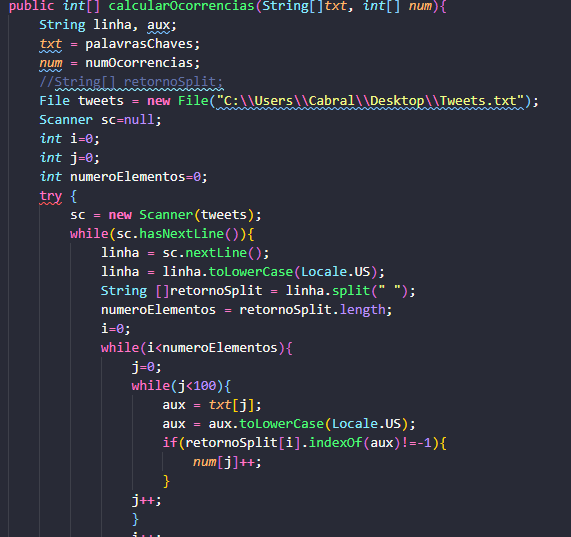
Eu adiciono “\n” no final para que fique gravado que acabou a linha e para ir para a próxima.

1. **Classe Tweets**

No método preenchePalavrasChaves() eu adiciono 100 palavras chaves a um vetor de String que será usado para realizar a busca pelo número de Ocorrência das palavras:



No método calcularOcorrencias() como o próprio nome já diz eu calculo o número de ocorrência de cada palavra no arquivo de Tweets:

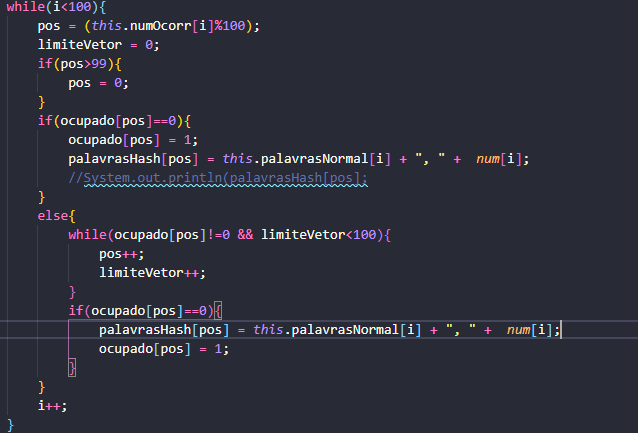


Eu abro o arquivo de Tweets e capturo linha por linha do arquivo, setto a linha inteira como minúscula para não dar erro na hora de fazer as comparações com o meu vetor de String que contém as palavras chaves. Utilizo o método “.split” para quebrar a linha no espaço, pegando assim palavra por palavra e poder comparar com as palavras chaves, posteriormente utilizo o retorno desse split em um indexOf com a minha palavra chave, caso o retorno seja diferente de -1 significa que há a ocorrência da palavra e posso incrementar a posição do vetor que contém o número de ocorrências.

Eu imprimo na main vetor de ocorrências chamado “ocorrencias” e o vetor de palavras chaves chamado “palavras”;

1. **Classe Hashing**

Nesta classe eu utilizo um vetor hashing para armazenamento das palavras chaves com seus respectivos números e ocorrências, para isso eu crio um vetor chamado “ocupado”, ele tem as mesmas dimensões que os outros vetores, e se inicia com 0, a partir do momento que eu for inserir um elemento o meu vetor “ocupado” vai para 1 na posição que for inserida no outro vetor, sendo assim o vetor “ocupado” é um vetor de controle, execução do vetor hashing:

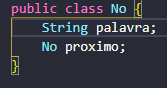


Com o vetor de controle “ocupado” e com o teste para que não haja mais que 100 interações no vetor, ou com o teste caso o contador chegue no final do vetor, eu redireciono ele para o começo, eu garanto que todas as 100 posições do vetor serão ocupadas, ainda há um teste de colisão, que joga para a próxima posição que estiver vazia. Eu concatenei as palavras chaves e o número de Ocorrências e imprimo tudo junto na Main com a variável chamada “palavrasHash”.

1. **Lista Simplesmente Encadeada**

**4.1 Classe No**

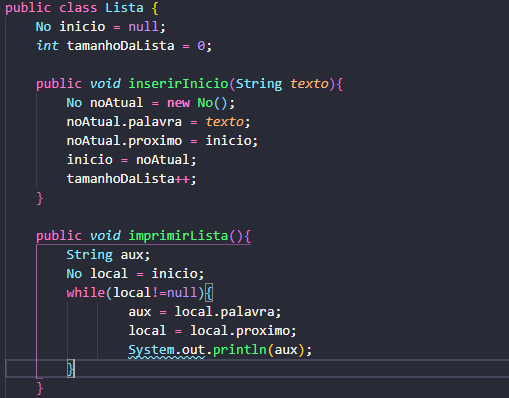
No caso da Lista Simplesmente Encadeada, há 2 classes, uma chamada “No”, na qual contém o campo de texto “palavra” que irá conter as palavras chaves concatenadas com o número de Ocorrências, esta concatenação foi realizada na main, mostrando a classe No:



**4.2 Classe Lista**

Nesta classe consta toda a manipulação de ponteiros da lista simplesmente encadeada, a criação do No de início e o tamanho da lista.

Há apenas 2 funções nessa classe, a função de inserir no início da lista e a função de imprimir a lista, como mostrado na imagem:

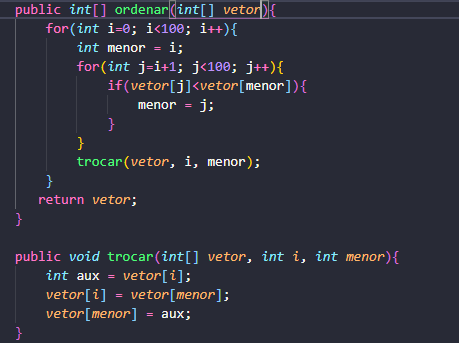


Não há muito segredo, o método de inserir eu utilizo na main dentro de um laço, no qual eu percorro a lista de palavras chaves e a lista de ocorrências e concateno os 2 no campo palavra do nó da lista, assim que funciona a função de inserir.

Na função de Imprimir a lista eu crio um nó local que eu vou utlizar para percorrer a lista e ir imprimindo os textos que constam em cada nó, tendo assim a minha lista de palavras chaves com suas respectivas ocorrências.

1. **Classe SelectSort**

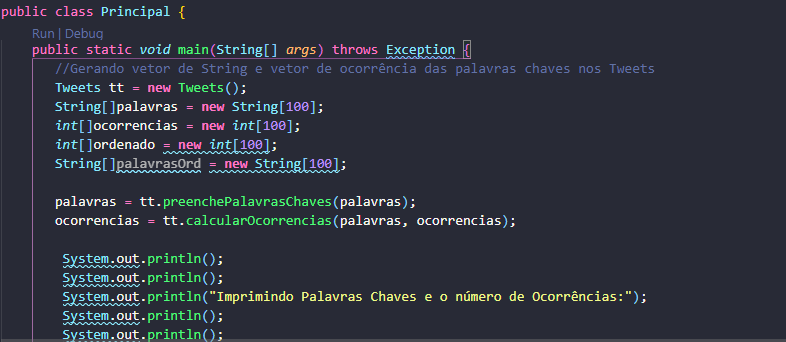
No começo do trabalho já imprimi a lista de palavras chaves com suas respectivas ocorrências, mas não estava ordenada, é aqui que entra essa classe, nessa classe eu faço a ordenação do número de ocorrências das palavras do menor para o maior utilizando o método SelectSort, como demonstrado na imagem:



Há 2 métodos nessa classe, um chamado “ordenar” e outro chamado “trocar”, dentro do método ordenar eu chamo o método trocar e só realizo a troca do número de ocorrências do vetor nele, é uma classe bem simples de se entender.

1. **Classe Principal**

A classe principal é a classe que contém a minha main, nela eu gero todos os objetos das outras classes, eu insiro e imprimo automaticamente cada um dos métodos criados, um abaixo do outro, na seguinte ordem: Palavras Chaves e suas ocorrências, Hashing, Lista Simplesmente Encadeada e por último o número de Ocorrências ordenado através do Selectsort, mostrando a classe Principal:



O trabalho funciona automaticamente, basta colocar o caminho do arquivo que contém o Tweets e ele automaticamente já inserirá em todas as estruturas e irá imprimir, dá de observar a utilização de cada uma através das mensagens de texto falando de cada uma pelo terminal.

1. **Conclusão**

Foi um trabalho bastante interessante e consegui ver a real diferença da implementação de cada estrutura, foi bem divertido implementar cada uma delas, espero que o trabalho tenha ficado bom, deu bastante trabalho, mas ao ver as estruturas funcionando como deveriam ser valeu tudo a pena, os conhecimentos aprendidos na implementação desse trabalho com certeza serão válidos para o resto da minha vida profissional.