a primeira implementação do programa foi fazer uma forma de ele conseguir ler o'que está no arquivo e poder separar em um array, primeira ideia foi

```
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.io.IOException;
import java.io.BufferedReader;
import java.io.FileReader;
public class Aug{
    public static void main(String[]args) throws FileNotFoundException
, IOException{
        FileReader arquivo = new FileReader("readme.txt");
        BufferedReader arquivobufado = new BufferedReader(arquivo);
        String linha = arquivobufado.readLine();
        System.out.println(linha);
        arquivobufado.close();
        String[] sublinha = linha.split(" ",1000);
        for(int i = 0; i < sublinha.length; i++){
            System.out.println("INDICE ["+i+"] = "+sublinha[i]);
        }


    }
}
```

Porém aqui já acontecia um problema, o programa só iria ler a primeira linha do código. Após algumas tentativas consegui entender o porque acontecia, e fui buscar uma forma de fazer a organização do vetor.

```
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.io.IOException;
import java.io.BufferedReader;
import java.io.FileReader;
public class Aug{
    public static void main(String[]args) throws FileNotFoundException
, IOException{
        FileReader arquivo = new FileReader("readme.txt");
        BufferedReader arquivobufado = new BufferedReader(arquivo);
        String linha;
        String allwords;
        String[] vetor =  {
            "Cachorro",
            "Banana",
            "Maçã",
```

```
            "Computador",
            "Abacaxi",
            "Elefante",
            "Dado",
            "Piano"
        };
        vetor = organizado(vetor);
        for(String item : vetor){
            System.out.println(item);
        }


    }public static String[] organizado (String[]vetor){
        String aux;
        for(int i = 0; i < vetor.length - 1; i++){
            for(int j = 0; j < vetor.length - i - 1; j++){
                if(vetor[j].compareTo(vetor[j + 1]) > 0){
                    aux = vetor[j];
                    vetor[j] = vetor[j+1];
                    vetor[j+1] = aux;
                }
            }
        }
        return vetor;
    }
}
```

Aqui estava fazendo testes para enxergar como iria funcionar a função para organizar o programa.

Após entender como iria funcionar a organização, parte para outro programa para conseguir fazer a busca binária enquanto era inserido as palavras, deixando de lado a organização.

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.BufferedReader;
import java.io.FileReader;

public class Aug3 {
    public static void main(String[] args) throws
FileNotFoundException, IOException {
        FileReader arquivo = new FileReader("readme.txt");
        BufferedReader arquivobufado = new BufferedReader(arquivo);
        String linha;
```

```
        String[] palavras = new String[1000];
        int indice = 0;

        while ((linha = arquivobufado.readLine()) != null) {
            String[] palavrasLinha = linha.toLowerCase().split(" ");
            for (String palavra : palavrasLinha) {
                if (indice < palavras.length) {
                    int bc = buscaBinaria(palavras, palavra, indice);
                    if (bc == -1) {
                        palavras[indice] = palavra;
                        indice++;
                    }
                } else {
                    System.out.println("Capacidade do vetor
excedida!");
                    break;
                }
            }
        }

        for (int i = 0; i < indice; i++) {
            System.out.println("INDICE [" + i + "] = " + palavras[i]);
        }

        arquivobufado.close();
    }

    public static int buscaBinaria(String[] array, String alvo, int
tamanho) {
        int esquerda = 0;
        int direita = tamanho - 1;

        while (esquerda <= direita) {
            int meio = esquerda + (direita - esquerda) / 2;
            int comparacao = alvo.compareTo(array[meio]);
            if (comparacao == 0) {
                return meio;
            }
            if (comparacao > 0) {
                esquerda = meio + 1;
            } else {
                direita = meio - 1;
            }
```

```
        }
        return -1;
    }
}
```

Após juntar as duas partes, notei que o programa continuava inserido palavras duplicadas, isto é, o programa checar se tinha palavras iguais, porém caso elas tivessem com a primeira letra em caixa alta, ele já a considerava como uma nova palavra, para consertar isso fiz a busca binária responder um valor booleano para verificar se estava a mesma palavra ou não. antes ele só iria fazer o seguinte teste.
A palavra Mãe está em uma posição do vetor? Ele percorria o vetor e encontrava a palavra mãe,e ignorava já que não era a palavra Mãe em si.

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.BufferedReader;
import java.io.FileReader;

public class Trabalho2 {
    public static void main(String[] args) throws
FileNotFoundException, IOException {
        FileReader arquivo = new FileReader("readme.txt");
        BufferedReader arquivobufado = new BufferedReader(arquivo);
        String linha;
        String[] palavras = new String[1000];
        int indice = 0;

        while ((linha = arquivobufado.readLine()) != null) {
            String[] palavrasLinha =
linha.toLowerCase().split("[\\s.,]+");
            for (String palavra : palavrasLinha) {
                if (indice < palavras.length) {
                    if (!contem(palavras, palavra, indice)) {
                        palavras[indice] = palavra;
                        indice++;
                    }
                } else {
                    System.out.println("Capacidade do vetor
excedida!");
                    break;
                }
            }
           palavras = organizado(palavras, indice);
        }
```

```
        for (int i = 0; i < indice; i++) {
            System.out.println("INDICE [" + i + "] = " + palavras[i]);
        }

        arquivobufado.close();
    }

    public static boolean contem(String[] array, String alvo, int
tamanho) {
        for (int i = 0; i < tamanho; i++) {
            if (array[i] != null && array[i].equalsIgnoreCase(alvo)) {
                return true;
            }
        }
        return false;
    }
    public static String[] organizado(String[] vetor, int tamanho) {
        String aux;
        for (int i = 0; i < tamanho - 1; i++) {
            for (int j = 0; j < tamanho - i - 1; j++) {
                if (vetor[j].compareTo(vetor[j + 1]) > 0) {
                    aux = vetor[j];
                    vetor[j] = vetor[j + 1];
                    vetor[j + 1] = aux;
                }
            }
        }
        return vetor;
    }
}
```