Lista 7 - Francisco Braz

1-)

a-)

```
algoritmo kpListaPadroes(T, n, k, m, P): vazio
início
    q := 3354393
    d := 32
    dM := 1
    para i = 1 até m-1 faça dM := (d*dM) mod q
    h2 := 0
    para i = 1 até m faça h2:= (h2*d + index(T[i])) mod q

    para j = 1 até k faça
    início
        recorrencias[j] = -1
        para i = 1 até m faça
        início
            h1Array[j] := (h1Array[j]*d + index(P[j][i])) mod q
        fim
    fim
    i := 1
    enquanto (i <= n-m) faça
    início
        para j = 1 até k faça
        início
            se h1Array[j] = h2 então
            início
                imprima “Padrão” + j + “ocorre em” + i
                recorrencias[j] = 1
            fim
        fim
        h2 := (h2+ d*q – index(T[i]*dM)) mod q
        h2 := (h2*d + index(T[i+m])) mod q
        i := i + 1;
    fim
    para j = 1 até k faça
        se recorrencias[j] = -1 então imprima “Padrão inexistente”
```

fim

b-)

Vamos considerar que temos uma boa função hash e não precisamos checar por falsos positivos. Nosso primeiro loop, que é responsável por calcular o valor de *dM* irá executar operações de tempo constante no máximo m-1 vezes. Portanto, temos:

$$\sum_{i=1}^{m-1} c = c * (m - 1 - 1 + 1) = c * (m - 1)$$

Posteriormente, temos o loop que calcula o hash dos *m* caracteres do texto. Esse loop irá executar operações de tempo constante no máximo *m* vezes. Logo:

$$\sum_{i=1}^{m} c = c * (m - 1 + 1) = c * m$$

Logo em seguida, temos os loops responsáveis por calcular o hash de cada padrão.

$$\sum_{j=1}^{k}\sum_{i=1}^{m} c = \sum_{j=1}^{k} c * (m - 1 + 1) = \sum_{j=1}^{k} c * m = c * m * k - 1 + 1) = c * m * k$$

Por último, temos a parte do algoritmo que faz a verificação entre o hash de cada padrão com o hash do texto. Nosso loop enquanto irá executar um total de *n-m,*logo::

$$\sum_{i=1}^{n-m}\sum_{j=1}^{k} c = \sum_{i=1}^{n-m} c * (k - 1 + 1) = \sum_{i=1}^{n-m} c * k = c * k * (n - m - 1 + 1) = c * k * (n - m)$$

Juntando todas essas complexidades, nós temos:

$$c * (m - 1) + c * m + c * m * k + c * (n - m) * k$$

Se ignorarmos as constantes:

$$m + m + m * k + (n - m) * k = m + m + mk + nk - mk$$

$= 2m + nk$

Portanto, nossa complexidade seria:
O(m+nk)

Se considerarmos que nosso algoritmo verificasse os falsos positivos seu pior caso (ocorrência de inúmeras colisões e necessidade de checar falsos positivos) na procura de 1 padrão é O(nm), então se fossemos procurar um total de *k* padrões iríamos ter $(n * m) * k$, logo, nossa complexidade seria: O(nmk)

c-) Implementação no arquivo kpListaPadroes.c