Introdução à Ciência de Computação II

Prova 1 – 07/10/2021

Lucas da Silva Claros N° 12682592

Questão 1:

```c
unsigned int* search_and_count(int** matriz, int* vetor, int m, int z) {
    int i, j, k, k2, el;
    unsigned int* con = calloc(z, sizeof(unsigned int));       // a (aritmética -> z * sizeof(unsigned int))
    for (k = 0; k < z; k++) {
        el = vetor[k];
        for(i = 0; i < m; i++){
        for(j = 0; j < m; j++){
            if (matriz[i][j] == el){                           // z * m^2 * c (para cada valor do vetor haverá comparações para cada elementro da matriz,
                                                               //             a matriz possui m^2 elementos e o vetor possui z elementos logo z* m^2 * c)
            k2 = 0;
            while (k2 < z) {                                   // (z + 1) * c (a comparação somente d owhile será feita desde k=0 até k=z,
                                                               //              logo z+1 comparações)
                                                               // todas as operações dentro do while serão executados z vezes no pior caso:
                if (vetor[k2] == el) {                         // z * c
                    con[k2]++;                                 // z * c
                }
                k2++;                                          // z * a
            }
            }
        }
        }
    }
    return con;
}
```

```c
/*
    A funcão dependerá do tamanho da matriz (m) e do tamanho do vetor (z):

    f(m,z) = a + z(m^2)c + zc + c + zc + zc + za

    mas precisamos em função de n. Sabemos que n = m^2 e que z < n/2, logo podemos aproximar para:

    f(n) = a + (n/2)*n*c + (n/2)c + c + (n/2)c + (n/2)a

    simplificando:

    f(n) = a + (n/2)(nc + 2c + a) + c
    f(n) = a + (n^2*c)/2 + (n * 2c)/2 + (n * a)/2 + c

*/
```

Questão 3:

```c
unsigned int* search_and_count_but_better(int** matriz, int* vetor, int m, int z) {
    int j, k, k2;
    unsigned int* con = calloc(z, sizeof(unsigned int));                    // a

    for (k = 0; k < z; k++)
    {
        for (k2 = 0; k2 < m; k2++)
        {
            if(vetor[k] < matriz[k2][k2] && k2 > 0)                         // (zm) * 2c
            {
                int i2 = (k2-1);                                            // (zm) * a

                // o primeiro for verificará na linha anterior, desde a coluna da diagonal anterior até a última coluna
                // o segundo verificará na linha da diagonal maior que o elemento analisado, desde a primeira coluna até a coluna da diagonal maior
                // percorrendo sempre o número de linhas + as duas diagonais -> (m+2) * c+a
                // no total -> (zm) * (m + 2) * (c + a)

                for (j = i2; j < m; j++)
                    if(matriz[i2][j] == vetor[k])
                        con[k]++;
                i2 = k2;
                for (j = 0; j < k2; j++)
                    if(matriz[i2][j] == vetor[k])
                        con[k]++;
                break;
            }
        }
    }
    return con;
}
```

```c
/*
    A funcão dependerá do tamanho da matriz (m) e do tamanho do vetor (z):

    f(m,z) = a + zm(2c + a + (m+2)(c+a))

    simplificando:
    f(m,z) = a + zm(2c + a + mc + ma + 2c + 2a)
    f(m,z) = a + zm(4c + 3a + m(c+a))

    mas precisamos em função de n. Sabemos que n = m^2 e que z < n/2, logo podemos aproximar para:

    zm = (n/2) * sqrt(n) = ((n^(3/2))/2)

    f(n) = a + ((n^(3/2))/2)*(4c + 3a + sqrt(n)*(c+a))

*/
```