PROVA 3 AAA - Vitória Melo

Questão 1

A complexidade do algoritmo é linear em n, visto que é somente um for de 0 até n.
O valor retornado está correto pois, utilizando uma ideia matemática, e gulosa, sabemos que uma sequência de M valores, para ser quebrada de forma a não possuir uma sequência k, precisa ser quebrada "no meio" de duas sequências de tamanho k. Dessa forma a divisão inteira M por K nos retorna esse valor.

Algoritmo(v[n], K): // n é 0-indexado

```
countImpar = 0
anterior = 0 // (não precisou ser usado)
Se v[0] % 2 == 1:
    countImpar = 1
    anterior = 1
for i = 1 até n:
    Se v[i] % 2 == 0:
        ans += countImpar Div K   → considere a divisão inteira
        countImpar = 0

    Senão
        countImpar ++

ans += countImpar Div k
ret ans
```

Fim-Algoritmo

PROVA 3 PAA - Nelson Melo

Questão 2

Sabendo que a entrada está ordenada [redacted] de forma estritamente crescente pelo final, podemos manter a melhor resposta como o melhor fim (menor fim) com maior contagem de vídeos assistidos.

```
Algoritmo(v, n):
    ansCurr = (0, 0) // [redacted] (Considere algo como pair do C++)

    for i = 1 até n: //pulando em n
    Se v[i] >= ans[redacted].second:
        ansCurr = (ansCurr.first + 1, i)

    return ansCurr.first

Fim-Algoritmo
```

PROVA 3 PAA - Welerson Melo

Questão 3

A ideia consiste em dividir ao meio, e na hora do "merge", retornar um array de 3 posições contendo os 3 maiores. Podemos comparar esta ideia ao MergeSort, alterando-se a quantidade de comparações a cada diferente altura da árvore de recursão, que em nosso caso é $\Theta(3)$ para cada nó, ou seja, $T(n) = 2T(n/2) + \Theta(3) \Rightarrow O(n)$ no pior caso.

Algoritmo

```
TME(v, l, r):
    Se (r == l):
        ret [v[r], -INF, -INF]      // considere -infinito
    Senão Se (r > l)
        ret [-INF, -INF, -INF]

    m = l + (r-1)/2
    a = TME(v, l, m)
    b = TME(v, m+1, r)

    resp = [0, 0, 0] // inicializando a resposta a retornar.
    i = j = 0
    for k de 0 até 3: // dos dados em 3
        Se (a[i] > b[j]):
            resp[k] = a[i++]
        Senão Se (b[j] > a[i])
            resp[k] = b[j++]

        Senão resp[k] = b[j++], i++;
```

continuação Questão 3

```
    (~~ret resp~~)
    ret resp
Fim TME;

Algoritmo(v):
    respFinal = TME(v, 0, len(v))

    ret respFinal[2]
FimAlgoritmo
```