Most frequent

Strategy

Input: An array A of n positive integers
Output: The most frequent number in A

```
1  Procedure MostFrequent1(A)
2      H ← new HashMap()
3      for x in A do
4          H.put(x, 0)
5      m ← A[0]
6      for x in A do
7          f ← H.get(x)
8          H.put(x, f + 1)
9          if H.get(x) > H.get(m) then
10             m ← x
11     return m
```

Strategy

Input: An array A of n positive integers
Output: The most frequent number in A

```
1  Procedure MostFrequent2(A)
2      k ← maximum element in A
3      B ← array of size k + 1
4      for i ← 0 to k do
5          B[i] ← 0
6      m ← A[0]
7      for x in A do
8          B[x] ← B[x] + 1
9          if B[x] > B[m] then
10             m ← x
11     return m
```

Above you see two (correct) algorithms that both take an array A of positive integers as input, and return the number that occurs most frequently. Discuss the advantages and disadvantages of the two different strategies. You should state both the run-time complexity in the *worst* case, and the *expected* run-time complexity, with respect to n and k. Justify your analysis *briefly*. You should also state in which situations **Strategy 1** should be preferred over **Strategy 2**, and vice versa, with regard to concrete running time. Be sure to clarify any assumptions you make.