1. In this problem we will examine several on-line algorithms for list access: MF (move-to-front), T (transpose), and FC (frequency count) on a specific request sequence. Consider a list \( x_1, x_2, \ldots, x_k \) with the following sequence \( k \) accesses to \( x_k \), \( k - 1 \) accesses to \( x_{k-1} \), all the way to 1 access to \( x_1 \). Altogether there are \( n = k(k + 1)/2 \) accesses with no insertions or deletions.

   (a) Calculate (as a function of \( k \)) the cost of MF, T, and FC for this request sequence.

   (b) Use your result to prove that T and FC are not constant competitive. Use the fact that MF is 2-competitive to achieve your result.

2. A generalization of the paging problem is called the \( k \)-server problem where we have a metric space \((M, d)\) of points and \( k \) servers which lie on \( k \) points. Recall a metric space \((M, d)\) has the property that \( d \) is a mapping from \( M \times M \) into the real numbers such that \( d(x, y) \geq 0 \), \( d(x, y) = 0 \) implies \( x = y \), \( d(x, y) = d(y, x) \), and \( d(x, z) \leq d(x, y) + d(y, z) \). A request is simply a member of \( M \). A request is said to be served if one of the servers is on the requested point. The cost of serving a request \( r \) is \( d(r, x) \) where the server on point \( x \) is moved to \( r \) to serve the request.

   (a) Define a metric space that makes the paging problem with cache size \( k \) into a \( k \)-server problem.

   (b) Consider the following metric space with exactly three points \( a, b \) and \( c \) on a line. The points \( a \) and \( c \) are 1 unit apart and \( b \) is between \( a \) and \( c \) exactly with distance 1/3 from \( a \) and 2/3 from \( c \). There are 2 servers. The greedy on-line algorithm always serves a request by moving the nearest server to it. Show that the greedy on-line algorithm is not constant competitive.