Design and Analysis of Algorithms

Problem #1.
Let ***T[1..n]*** be a *sorted* array of *distinct* integers, some of which may be negative. Give an algorithm that can find an index ***i*** such that $1 \leq i \leq n$ and ***T[i]=i***, provided such an index exists. Your algorithm should take a time in $O(\log n)$ in the worst case.

Problem #2.
There are ***n*** trading posts along a river. At any of the post you can rent a canoe to be returned at any other post downstream. For each possible departure point ***i*** and each possible arrival point ***j,*** the cost of a rental from ***i*** to ***j*** is known. However, it can happen that the cost of renting from ***i*** to ***j*** is higher than the total cost of a series of shorter rentals. In this case you can return the first canoe at some post ***k*** between ***i*** and ***j*** and continue your journey in a second canoe. There is no extra charge for changing canoes in this way. Give an efficient algorithm to determine the minimum cost of a trip by canoe from each possible departure point ***i*** to each possible arrival point ***j***. In terms of ***n***, how much time is needed by your algorithm?

Problem #3.
Demonstrate the insertion of the keys 4,27,18,14,19,32,11,16 into a hash table with collisions resolved by *linear probing*. Assume that the table has 9 slots and that the hash function is $h(k) = k \bmod 9$. Draw the state of the hash table after all insertions.
Propose strategies other than *linear probing* for handling *collisions* in a hash table. Are they better, worse?