Exercise 1: Hashing - Collision Resolution with Open Addressing

(a) Let \( h(s, j) := h_1(s) - 2j \mod m \) and let \( h_1(x) = x + 2 \mod m \). Insert the keys 51, 13, 21, 30, 23, 72 into the hash table of size \( m = 7 \) using linear probing for collision resolution (the table should show the final state).

(b) Let \( h(s, j) := h_1(s) + j \cdot h_2(s) \mod m \) and let \( h_1(x) = x \mod m \) and \( h_2(x) = 1 + (x \mod (m - 1)) \). Insert the keys 28, 59, 47, 13, 39, 69, 12 into the hash table of size \( m = 11 \) using the double hashing probing technique for collision resolution. The hash table below should show the final state.

Exercise 2: Application of Hashtables

Consider the following algorithm:

```
Algorithm 1 algorithm
1: for i = 1 to n - 1 do
2:   for j = 0 to i - 1 do
3:     for k = 0 to n - 1 do
5:         return true
6: return false
```

(a) Describe what \texttt{algorithm} computes and analyse its asymptotical runtime.

(b) Describe a different algorithm \( B \) for this problem (i.e., \( B(A) = \text{algorithm}(A) \) for each input \( A \)) which uses hashing and takes time \( \mathcal{O}(n^2) \).

\textit{You may assume that inserting and finding keys in a hash table needs \( \mathcal{O}(1) \) if \( \alpha = \mathcal{O}(1) \) (\( \alpha \) is the load of the table).}

(c) Describe another algorithm for this problem without using hashing which takes time \( \mathcal{O}(n^2 \log n) \).