2) (10 pts) ALG (Heaps/Hash Tables)

Consider the following hash table and the strings it already contains, along with the hash function being used to insert strings into the table. Assume the table only stores alphabetic strings.

Note: The length of the hash table is **11**.

llama	xenon	want	yurt		mop	nook			uvula	
0	1	2	3	4	5	6	7	8	9	10

```
// This function (which is pretty bad for hashing strings in the real
// world, by the way) assumes str is non-NULL and non-empty.
int hash(char *str)
{
   // Note: This converts letters on the range 'a' through 'z' or
   // 'A' through 'Z' to integers on the range 0 through 25.
   // For example: 'a' -> 0, 'b' -> 1, ..., 'z' -> 25.
   return (tolower(str[0]) - 'a')%11;
}
```

For each of the following questions, refer to the original hash table above. For example, in part (b), refer to the original table – **not** the table that contains the string you come up with in part (a).

a. (2 pts) Give a string that, if **inserted** into the table above using **quadratic probing**, would cause us to encounter the **minimum** number of collisions possible.

Any string starting with: e, h, i, k, p, s, t, or v (… which lead to cells 4, 7, 8, and 10)
Grading: all or nothing

b. (2 pts) Give a string that, if **inserted** into the table above using **quadratic probing**, would cause us to encounter the **maximum** number of collisions possible.

Any string starting with: c, f, n, q, or y (… which lead to cells 2 and 5) **Grading:** all or nothing

c. (5 pts) Give all the alphabetic letters someone could have used to start their string in order to give a correct answer for part (b) of this problem.

c, f, n, q, y (no need to also list uppercase letters) **Grading:** 1 pt each

d. (1 pt) Give a string that, if inserted into the table above using **linear probing**, would cause us to encounter the **maximum** number of collisions possible.

Any string starting with: a, l, or w (… which lead to cell 0) **Grading:** all or nothing