```
1 #include <cs50.h>
    #include <stdio.h>
    #include <stdlib.h>
    #include <string.h>
 5
 6
    typedef struct node
 7
 8
        string phrase;
 9
        struct node *next;
    } node;
10
11
12
    #define LIST SIZE 2
13
    bool unload(node *list);
14
15
    void visualizer(node *list);
16
17
    int main(void)
18
    {
        node *list = NULL;
19
20
21
        // Add items to list
22
        for (int i = 0; i < LIST SIZE; i++)
23
        {
             string phrase = get string("Enter a new phrase: ");
24
25
            // TODO: add phrase to new node in list
26
27
28
            // Visualize list after adding a node.
29
            visualizer(list);
        }
30
31
32
        // Free all memory used
33
        if (!unload(list))
34
35
             printf("Error freeing the list.\n");
36
             return 1;
        }
37
38
        printf("Freed the list.\n");
39
40
        return 0;
41
    }
42
```

```
43
    bool unload(node *list)
44
45
        // TODO: Free all allocated nodes
        return false;
46
47
    }
48
49
    void visualizer(node *list)
50
51
        printf("\n+-- List Visualizer --+\n\n");
52
        while (list != NULL)
53
54
            printf("Location %p\nPhrase: \"%s\"\nNext: %p\n\n", list, list->phrase, list->next);
55
           list = list->next;
56
        printf("+----+\n\n");
57
58
    }
```

```
1 #include <cs50.h>
2 #include <stdio.h>
    #include <stdlib.h>
    #include <string.h>
    typedef struct node
 6
 7
 8
        string phrase;
 9
        struct node *next;
    } node;
10
11
12
    node *table[26];
13
    int hash(string phrase);
14
15
    bool unload(node *list);
16
    void visualizer(node *list);
17
18
    int main(void)
19
    {
20
        // Add items
        for (int i = 0; i < 3; i++)
21
22
23
            string phrase = get string("Enter a new phrase: ");
24
25
            // Find phrase bucket
26
            int bucket = hash(phrase);
27
            printf("%s hashes to %i\n", phrase, bucket);
28
        }
29
    }
30
31
    // TODO: return the correct bucket for a given phrase
32
    int hash(string phrase)
33
    {
34
        return 0;
35
    }
```