```c
// Implements linear search for integers

#include <cs50.h>
#include <stdio.h>

int main(void)
{
    // An array of integers
    int numbers[] = {20, 500, 10, 5, 100, 1, 50};

    // Search for number
    int n = get_int("Number: ");
    for (int i = 0; i < 7; i++)
    {
        if (numbers[i] == n)
        {
            printf("Found\n");
            return 0;
        }
    }
    printf("Not found\n");
    return 1;
}
```

```c
1   // Implements linear search for strings
2
3   #include <cs50.h>
4   #include <stdio.h>
5   #include <string.h>
6
7   int main(void)
8   {
9       // An array of strings
10      string strings[] = {"battleship", "boot", "cannon", "iron", "thimble", "top hat"};
11
12      // Search for string
13      string s = get_string("String: ");
14      for (int i = 0; i < 6; i++)
15      {
16          if (strcmp(strings[i], s) == 0)
17          {
18              printf("Found\n");
19              return 0;
20          }
21      }
22      printf("Not found\n");
23      return 1;
24  }
```

```c
// Implements a phone book without structs

#include <cs50.h>
#include <stdio.h>
#include <string.h>

int main(void)
{
    // Arrays of strings
    string names[] = {"Carter", "David", "John"};
    string numbers[] = {"+1-617-495-1000", "+1-617-495-1000", "+1-949-468-2750"};

    // Search for name
    string name = get_string("Name: ");
    for (int i = 0; i < 3; i++)
    {
        if (strcmp(names[i], name) == 0)
        {
            printf("Found %s\n", numbers[i]);
            return 0;
        }
    }
    printf("Not found\n");
    return 1;
}
```

```c
 1    // Implements a phone book with structs
 2
 3    #include <cs50.h>
 4    #include <stdio.h>
 5    #include <string.h>
 6
 7    typedef struct
 8    {
 9        string name;
10        string number;
11    }
12    person;
13
14    int main(void)
15    {
16        person people[3];
17
18        people[0].name = "Carter";
19        people[0].number = "+1-617-495-1000";
20
21        people[1].name = "David";
22        people[1].number = "+1-617-495-1000";
23
24        people[2].name = "John";
25        people[2].number = "+1-949-468-2750";
26
27        // Search for name
28        string name = get_string("Name: ");
29        for (int i = 0; i < 3; i++)
30        {
31            if (strcmp(people[i].name, name) == 0)
32            {
33                printf("Found %s\n", people[i].number);
34                return 0;
35            }
36        }
37        printf("Not found\n");
38        return 1;
39    }
```

```c
// Draws a pyramid using iteration

#include <cs50.h>
#include <stdio.h>

void draw(int n);

int main(void)
{
    // Get height of pyramid
    int height = get_int("Height: ");

    // Draw pyramid
    draw(height);
}

void draw(int n)
{
    // Draw pyramid of height n
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < i + 1; j++)
        {
            printf("#");
        }
        printf("\n");
    }
}
```

```c
// Draws a pyramid (incorrectly) using recursion,
// must be compiled with -Wno-infinite-recursion

#include <cs50.h>
#include <stdio.h>

void draw(int n);

int main(void)
{
    draw(1);
}

void draw(int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("#");
    }
    printf("\n");

    draw(n + 1);
}
```

```c
1    // Draws a pyramid using recursion
2
3    #include <cs50.h>
4    #include <stdio.h>
5
6    void draw(int n);
7
8    int main(void)
9    {
10       // Get height of pyramid
11       int height = get_int("Height: ");
12
13       // Draw pyramid
14       draw(height);
15   }
16
17   void draw(int n)
18   {
19       // If nothing to draw
20       if (n <= 0)
21       {
22           return;
23       }
24
25       // Draw pyramid of height n - 1
26       draw(n - 1);
27
28       // Draw one more row of width n
29       for (int i = 0; i < n; i++)
30       {
31           printf("#");
32       }
33       printf("\n");
34   }
```