```c
// Prints two strings' addresses

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

int main(void)
{
    // Get two strings
    string s = get_string("s: ");
    string t = get_string("t: ");

    // Print strings' addresses
    printf("s: %p\n", s);
    printf("t: %p\n", t);
}
```

```c
// Compares two integers

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

int main(void)
{
    // Get two integers
    int i = get_int("i: ");
    int j = get_int("j: ");

    // Compare integers
    if (i == j)
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
}
```

```c
// Compares two strings' addresses

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

int main(void)
{
    // Get two strings
    string s = get_string("s: ");
    string t = get_string("t: ");

    // Compare strings' addresses
    if (s == t)
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
}
```

```c
// Compares two strings for equality

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

bool compare_strings(string a, string b);

int main(void)
{
    // Get two strings
    string s = get_string("s: ");
    string t = get_string("t: ");

    // Compare strings for equality
    if (compare_strings(s, t))
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
}

bool compare_strings(string a, string b)
{
    // Compare strings' lengths
    if (strlen(a) != strlen(b))
    {
        return false;
    }

    // Compare strings character by character
    for (int i = 0, n = strlen(a); i < n; i++)
    {
        // Different
        if (a[i] != b[i])
        {
            return false;
        }
    }

    // Same
```

```
45        return true;
46    }
```

```c
// Compares two strings for equality

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

bool compare_strings(char *a, char *b);

int main(void)
{
    // Get two strings
    char *s = get_string("s: ");
    char *t = get_string("t: ");

    // Compare strings for equality
    if (compare_strings(s, t))
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
}

bool compare_strings(char *a, char *b)
{
    // Compare strings' lengths
    if (strlen(a) != strlen(b))
    {
        return false;
    }

    // Compare strings character by character
    for (int i = 0, n = strlen(a); i < n; i++)
    {
        // Different
        if (a[i] != b[i])
        {
            return false;
        }
    }

    // Same
```

```
45          return true;
46      }
```

```c
// Compares two strings for equality using strcmp

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

int main(void)
{
    // Get two strings
    char *s = get_string("s: ");
    char *t = get_string("t: ");

    // Compare strings for equality
    if (strcmp(s, t) == 0)
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
}
```

```c
// Compares two strings for equality while checking for errors

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

int main(void)
{
    // Get a string
    char *s = get_string("s: ");
    if (s == NULL)
    {
        return 1;
    }

    // Get another string
    char *t = get_string("t: ");
    if (t == NULL)
    {
        return 1;
    }

    // Compare strings for equality
    if (strcmp(s, t) == 0)
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
    return 0;
}
```

```c
// Compares two strings for equality while checking (succinctly) for errors

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

int main(void)
{
    // get a string
    char *s = get_string("s: ");
    if (!s)
    {
        return 1;
    }

    // get another string
    char *t = get_string("t: ");
    if (!t)
    {
        return 1;
    }

    // compare strings for equality
    if (strcmp(s, t) == 0)
    {
        printf("same\n");
    }
    else
    {
        printf("different\n");
    }
    return 0;
}
```

```c
// Capitalizes a string

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

int main(void)
{
    // Get a string
    string s = get_string("s: ");

    // Copy string's address
    string t = s;

    // Capitalize first letter in string
    if (strlen(t) > 0)
    {
        t[0] = toupper(t[0]);
    }

    // Print string twice
    printf("s: %s\n", s);
    printf("t: %s\n", t);
}
```

```c
// Capitalizes a copy of a string while checking for errors

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

int main(void)
{
    // Get a string
    char *s = get_string("s: ");
    if (!s)
    {
        return 1;
    }

    // Allocate memory for another string
    char *t = malloc((strlen(s) + 1) * sizeof(char));
    if (!t)
    {
        return 1;
    }

    // Copy string into memory
    for (int i = 0, n = strlen(s); i <= n; i++)
    {
        t[i] = s[i];
    }

    // Capitalize first letter in copy
    if (strlen(t) > 0)
    {
        t[0] = toupper(t[0]);
    }

    // Print strings
    printf("s: %s\n", s);
    printf("t: %s\n", t);

    // Free memory
    free(t);
    return 0;
}
```

```c
// Capitalizes a copy of a string using strcpy while checking for errors

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

int main(void)
{
    // Get a string
    char *s = get_string("s: ");
    if (!s)
    {
        return 1;
    }

    // Allocate memory for another string
    char *t = malloc((strlen(s) + 1) * sizeof(char));
    if (!t)
    {
        return 1;
    }

    // Copy string into memory
    strcpy(t, s);

    // Capitalize first letter in copy
    if (strlen(t) > 0)
    {
        t[0] = toupper(t[0]);
    }

    // Print strings
    printf("s: %s\n", s);
    printf("t: %s\n", t);

    // Free memory
    free(t);
    return 0;
}
```

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

int main(void)
{
    string name = get_string("Name: ");
    printf("hello, %s\n", name);
}
```

```c
// http://valgrind.org/docs/manual/quick-start.html#quick-start.prepare

#include <stdlib.h>

void f(void)
{
    int *x = malloc(10 * sizeof(int));
    x[10] = 0;
}

int main(void)
{
    f();
    return 0;
}
```

```c
// Fails to swap two integers

#include <stdio.h>

void swap(int a, int b);

int main(void)
{
    int x = 1;
    int y = 2;

    printf("x is %i, y is %i\n", x, y);
    swap(x, y);
    printf("x is %i, y is %i\n", x, y);
}

void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
```

```c
// Gets an int from user using scanf

#include <stdio.h>

int main(void)
{
    int x;
    printf("x: ");
    scanf("%i", &x);
    printf("x: %i\n", x);
}
```

```c
// Incorrectly gets a string from user using scanf

#include <stdio.h>

int main(void)
{
    char *s;
    printf("s: ");
    scanf("%s", s);
    printf("s: %s\n", s);
}
```

```c
// Dangerously gets a string from user using scanf

#include <stdio.h>

int main(void)
{
    char s[5];
    printf("s: ");
    scanf("%s", s);
    printf("s: %s\n", s);
}
```

```c
// Prints a string's characters using square brackets

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

int main(void)
{
    // Get a string
    char *s = get_string("string: ");
    if (!s)
    {
        return 1;
    }

    // Print string, one character per line
    for (int i = 0, n = strlen(s); i < n; i++)
    {
        printf("%c\n", s[i]);
    }
    return 0;
}
```

```c
// Prints a string's characters using pointer arithmetic

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

int main(void)
{
    // Get a string
    char *s = get_string("string: ");
    if (!s)
    {
        return 1;
    }

    // Print string, one character per line
    for (int i = 0, n = strlen(s); i < n; i++)
    {
        printf("%c\n", *(s + i));
    }
    return 0;
}
```

```c
// Demonstrates lack of structs

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

int main(void)
{
    // Space for students
    int enrollment = get_int("Enrollment: ");
    string names[enrollment];
    string dorms[enrollment];

    // Prompt for students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        names[i] = get_string("Name: ");
        dorms[i] = get_string("Dorm: ");
    }

    // Print students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        printf("%s is in %s.\n", names[i], dorms[i]);
    }
}
```

```c
// Demonstrates structs

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

#include "struct.h"

int main(void)
{
    // Space for students
    int enrollment = get_int("Enrollment: ");
    student students[enrollment];

    // Prompt for students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        students[i].name = get_string("Name: ");
        students[i].dorm = get_string("Dorm: ");
    }

    // Print students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        printf("%s is in %s.\n", students[i].name, students[i].dorm);
    }
}
```

```c
// Demonstrates file I/O

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

#include "struct.h"

int main(void)
{
    // Space for students
    int enrollment = get_int("Enrollment: ");
    student students[enrollment];

    // Prompt for students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        students[i].name = get_string("Name: ");
        students[i].dorm = get_string("Dorm: ");
    }

    // Save students to disk
    FILE *file = fopen("students.csv", "w");
    if (file)
    {
        for (int i = 0; i < enrollment; i++)
        {
            fprintf(file, "%s,%s\n", students[i].name, students[i].dorm);
        }
        fclose(file);
    }
}
```

```c
// Represents a student

typedef struct
{
    char *name;
    char *dorm;
}
student;
```

```c
// Swaps two integers using pointers

#include <stdio.h>

void swap(int *a, int *b);

int main(void)
{
    int x = 1;
    int y = 2;

    printf("x is %i, y is %i\n", x, y);
    swap(&x, &y);
    printf("x is %i, y is %i\n", x, y);
}

void swap(int *a, int *b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```