```
1   // Prints an integer
2
3   #include <stdio.h>
4
5   int main(void)
6   {
7       int n = 50;
8       printf("%i\n", n);
9   }
```

```c
1   // Prints an integer's address
2
3   #include <stdio.h>
4
5   int main(void)
6   {
7       int n = 50;
8       printf("%p\n", &n);
9   }
```

```
1   // Stores and prints an integer's address
2
3   #include <stdio.h>
4
5   int main(void)
6   {
7       int n = 50;
8       int *p = &n;
9       printf("%p\n", p);
10  }
```

```c
// Stores and prints an integer via its address

#include <stdio.h>

int main(void)
{
    int n = 50;
    int *p = &n;
    printf("%i\n", *p);
}
```

```
 1    // Prints a string
 2
 3    #include <cs50.h>
 4    #include <stdio.h>
 5
 6    int main(void)
 7    {
 8        string s = "HI!";
 9        printf("%s\n", s);
10    }
```

```
 1   // Prints address of first char of a string
 2
 3   #include <cs50.h>
 4   #include <stdio.h>
 5
 6   int main(void)
 7   {
 8       string s = "HI!";
 9       char c = s[0];
10       char *p = &c;
11       printf("%p\n", p);
12   }
```

```
1   // Prints address of first char of a string
2
3   #include <cs50.h>
4   #include <stdio.h>
5
6   int main(void)
7   {
8       string s = "HI!";
9       char *p = &s[0];
10      printf("%p\n", p);
11  }
```

```
 1   // Prints address of a string
 2
 3   #include <cs50.h>
 4   #include <stdio.h>
 5
 6   int main(void)
 7   {
 8       string s = "HI!";
 9       printf("%p\n", s);
10   }
```

```c
// Prints a string's address as well the addresses of its chars

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

int main(void)
{
    string s = "HI!";
    printf("%p\n", s);
    printf("%p\n", &s[0]);
    printf("%p\n", &s[1]);
    printf("%p\n", &s[2]);
    printf("%p\n", &s[3]);
}
```

```
1   // Prints a string's chars
2
3   #include <cs50.h>
4   #include <stdio.h>
5
6   int main(void)
7   {
8       string s = "HI!";
9       printf("%c\n", s[0]);
10      printf("%c\n", s[1]);
11      printf("%c\n", s[2]);
12  }
```

```c
// Stores and prints a string without using the CS50 Library

#include <stdio.h>

int main(void)
{
    char *s = "HI!";
    printf("%c\n", s[0]);
    printf("%c\n", s[1]);
    printf("%c\n", s[2]);
}
```

```c
// Stores and prints a string's address via pointer arithmetic

#include <stdio.h>

int main(void)
{
    char *s = "HI!";
    printf("%c\n", *s);
    printf("%c\n", *(s + 1));
    printf("%c\n", *(s + 2));
}
```

```c
// Prints an array using pointer arithmetic

#include <stdio.h>

int main(void)
{
    // An array of numbers
    int numbers[] = {4, 6, 8, 2, 7, 5, 0};

    // Print numbers
    printf("%i\n", *numbers);
    printf("%i\n", *(numbers + 1));
    printf("%i\n", *(numbers + 2));
    printf("%i\n", *(numbers + 3));
    printf("%i\n", *(numbers + 4));
    printf("%i\n", *(numbers + 5));
    printf("%i\n", *(numbers + 6));
}
```

```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
    char *s = get_string("s: ");
    char *t = get_string("t: ");

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

```c
// Compares two strings 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
    if (strcmp(s, t) == 0)
    {
        printf("Same\n");
    }
    else
    {
        printf("Different\n");
    }
}
```

```c
// Prints two strings

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

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

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

```c
// Prints two strings' addresses

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

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

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

```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
 1   // Capitalizes a copy of a string
 2
 3   #include <cs50.h>
 4   #include <ctype.h>
 5   #include <stdio.h>
 6   #include <stdlib.h>
 7   #include <string.h>
 8
 9   int main(void)
10   {
11       // Get a string
12       char *s = get_string("s: ");
13
14       // Allocate memory for another string
15       char *t = malloc(strlen(s) + 1);
16
17       // Copy string into memory
18       for (int i = 0, n = strlen(s); i <= n; i++)
19       {
20           t[i] = s[i];
21       }
22
23       // Capitalize copy
24       t[0] = toupper(t[0]);
25
26       // Print strings
27       printf("s: %s\n", s);
28       printf("t: %s\n", t);
29   }
```

```
 1    // Capitalizes a copy of a string using strcpy
 2
 3    #include <cs50.h>
 4    #include <ctype.h>
 5    #include <stdio.h>
 6    #include <stdlib.h>
 7    #include <string.h>
 8
 9    int main(void)
10    {
11        // Get a string
12        char *s = get_string("s: ");
13
14        // Allocate memory for another string
15        char *t = malloc(strlen(s) + 1);
16
17        // Copy string into memory
18        strcpy(t, s);
19
20        // Capitalize copy
21        t[0] = toupper(t[0]);
22
23        // Print strings
24        printf("s: %s\n", s);
25        printf("t: %s\n", t);
26    }
```

```c
// Capitalizes a copy of a string without memory errors

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

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

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

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

    // Capitalize 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
// Demonstrates memory errors via valgrind

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

int main(void)
{
    int *x = malloc(3 * sizeof(int));
    x[1] = 72;
    x[2] = 73;
    x[3] = 33;
}
```

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

int main(void)
{
    int scores[3];
    for (int i = 0; i < 3; i++)
    {
        printf("%i\n", scores[i]);
    }
}
```

```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
// 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;
}
```

```
1   // Gets an int from user using scanf
2
3   #include <stdio.h>
4
5   int main(void)
6   {
7       int x;
8       printf("x: ");
9       scanf("%i", &x);
10      printf("x: %i\n", x);
11  }
```

```c
 1   // Incorrectly gets a string from user using scanf
 2
 3   #include <stdio.h>
 4
 5   int main(void)
 6   {
 7       char *s;
 8       printf("s: ");
 9       scanf("%s", s);
10       printf("s: %s\n", s);
11   }
```

```
 1   // Dangerously gets a string from user using scanf
 2
 3   #include <stdio.h>
 4
 5   int main(void)
 6   {
 7       char s[4];
 8       printf("s: ");
 9       scanf("%s", s);
10       printf("s: %s\n", s);
11   }
```

```
1  name,number
```

```
 1    // Saves names and numbers to a CSV file
 2
 3    #include <cs50.h>
 4    #include <stdio.h>
 5    #include <string.h>
 6
 7    int main(void)
 8    {
 9        // Open CSV file
10        FILE *file = fopen("phonebook.csv", "a");
11        if (!file)
12        {
13            return 1;
14        }
15
16        // Get name and number
17        string name = get_string("Name: ");
18        string number = get_string("Number: ");
19
20        // Print to file
21        fprintf(file, "%s,%s\n", name, number);
22
23        // Close file
24        fclose(file);
25    }
```

```c
// Detects if a file is a JPEG

#include <stdint.h>
#include <stdio.h>

typedef uint8_t BYTE;

int main(int argc, char *argv[])
{
    // Check usage
    if (argc != 2)
    {
        return 1;
    }

    // Open file
    FILE *file = fopen(argv[1], "r");
    if (!file)
    {
        return 1;
    }

    // Read first three bytes
    BYTE bytes[3];
    fread(bytes, sizeof(BYTE), 3, file);

    // Check first three bytes
    if (bytes[0] == 0xff && bytes[1] == 0xd8 && bytes[2] == 0xff)
    {
        printf("Yes, possibly\n");
    }
    else
    {
        printf("No\n");
    }

    // Close file
    fclose(file);
}
```

```c
#include "helpers.h"

// Only let red through
void filter(int height, int width, RGBTRIPLE image[height][width])
{
    // Loop over all pixels
    for (int i = 0; i < height; i++)
    {
        for (int j = 0; j < width; j++)
        {
            image[i][j].rgbtBlue = 0x00;
            image[i][j].rgbtGreen = 0x00;
        }
    }
}
```