```
../README.txt

addresses*.c

compare*.c
copy*.c

memory.c
garbage.c

swap*.c

get*.c

phonebook*.c
cp.c
```

```c
// Prints an integer

#include <stdio.h>

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

```
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   }
```

```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   // Declares a string with CS50 Library
 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   // Declares a string without CS50 Library
2
3   #include <stdio.h>
4
5   int main(void)
6   {
7       char *s = "HI!";
8       printf("%s\n", s);
9   }
```

```c
// Prints a string's chars

#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
// Prints a string's chars 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 substrings via pointer arithmetic

#include <stdio.h>

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

```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
1    // Prints two strings' addresses
2
3    #include <cs50.h>
4    #include <stdio.h>
5
6    int main(void)
7    {
8        // Get two strings
9        char *s = get_string("s: ");
10       char *t = get_string("t: ");
11
12       // Print strings' addresses
13       printf("%p\n", s);
14       printf("%p\n", t);
15   }
```

```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
    t[0] = toupper(t[0]);

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

copy1.c

```c
// Capitalizes a string, checking length first

#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

#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: ");

    // Allocate memory for another string
    char *t = malloc(strlen(s) + 1);

    // Copy string into memory, including '\0'
    for (int i = 0; i <= strlen(s); i++)
    {
        t[i] = s[i];
    }

    // Capitalize copy
    t[0] = toupper(t[0]);

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

```c
// Capitalizes a copy of a string, defining n in loop too

#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: ");

    // Allocate memory for another string
    char *t = malloc(strlen(s) + 1);

    // Copy string into memory, including '\0'
    for (int i = 0, n = strlen(s); i <= n; i++)
    {
        t[i] = s[i];
    }

    // Capitalize copy
    t[0] = toupper(t[0]);

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

```c
// Capitalizes a copy of a string using strcpy

#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: ");

    // Allocate memory for another string
    char *t = malloc(strlen(s) + 1);

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

    // Capitalize copy
    t[0] = toupper(t[0]);

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

```c
 1   // Capitalizes a copy of a string without memory errors
 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       if (s == NULL)
14       {
15           return 1;
16       }
17
18       // Allocate memory for another string
19       char *t = malloc(strlen(s) + 1);
20       if (t == NULL)
21       {
22           return 1;
23       }
24
25       // Copy string into memory
26       strcpy(t, s);
27
28       // Capitalize copy
29       if (strlen(t) > 0)
30       {
31           t[0] = toupper(t[0]);
32       }
33
34       // Print strings
35       printf("s: %s\n", s);
36       printf("t: %s\n", t);
37
38       // Free memory
39       free(t);
40       return 0;
41   }
```

```
 1   // Demonstrates memory errors via valgrind
 2
 3   #include <stdio.h>
 4   #include <stdlib.h>
 5
 6   int main(void)
 7   {
 8       int *x = malloc(3 * sizeof(int));
 9       x[1] = 72;
10       x[2] = 73;
11       x[3] = 33;
12   }
```

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

int main(void)
{
    int scores[1024];
    for (int i = 0; i < 1024; 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;
}
```

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

#include <stdio.h>

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

```c
// Incorrectly gets a string from user using scanf; compile with -Wno-uninitialized

#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 with array

#include <stdio.h>

int main(void)
{
    char s[4];
    printf("s: ");
    scanf("%s", s);
    printf("s: %s\n", s);
}
```

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

#include <stdio.h>

int main(void)
{
    char s[4];
    printf("s: ");
    scanf("%s", s);
    printf("s: %s\n", s);
}
```

```c
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
12       // Get name and number
13       char *name = get_string("Name: ");
14       char *number = get_string("Number: ");
15
16       // Print to file
17       fprintf(file, "%s,%s\n", name, number);
18
19       // Close file
20       fclose(file);
21   }
```

```c
 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        char *name = get_string("Name: ");
18        char *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
1    // Copies a file
2
3    #include <stdio.h>
4    #include <stdint.h>
5
6    typedef uint8_t BYTE;
7
8    int main(int argc, char *argv[])
9    {
10       FILE *src = fopen(argv[1], "rb");
11       FILE *dst = fopen(argv[2], "wb");
12
13       BYTE b;
14
15       while (fread(&b, sizeof(b), 1, src) != 0)
16       {
17           fwrite(&b, sizeof(b), 1, dst);
18       }
19
20       fclose(dst);
21       fclose(src);
22   }
```