```
// Compara dois inteiros
 1
 3
4
     #include <cs50.h>
     #include <stdio.h>
 5
6
7
     int main(void)
 8
          // Recebe dois inteiros
          int i = get_int("i: ");
int j = get_int("j: ");
 9
10
11
          // Compara inteiros
12
13
          if (i == j)
14
          {
15
              printf("Iguais\n");
          }
else
16
17
18
          {
19
              printf("Diferentes\n");
20
21
     }
```

```
// Compara os endereços de duas strings
1
 3
4
     #include <cs50.h>
     #include <stdio.h>
 5
 6
7
     int main(void)
8
         // Recebe duas strings
         string s = get_string("s: ");
string t = get_string("t: ");
9
10
11
12
         // Compara os endereços das strings
13
         if (s == t)
14
         {
15
              printf("Iguais\n");
          }
16
17
         else
18
          {
19
              printf("Diferentes\n");
20
21
     }
```

```
// Imprime duas strings
 1
2
3
4
5
6
7
8
       #include <cs50.h>
       #include <stdio.h>
       int main(void)
             // Recebe duas strings
             string s = get_string("s: ");
string t = get_string("t: ");
 9
10
11
            // Imprime strings
printf("%s\n", s);
printf("%s\n", t);
12
13
14
15
      }
```

```
// Imprime os endereços de duas strings
 1
 3
4
      #include <cs50.h>
      #include <stdio.h>
 5
6
7
      int main(void)
 8
           // Recebe duas strings
            string s = get_string("s: ");
string t = get_string("t: ");
 9
10
11
           // Imprime os endereços das strings
printf("%p\n", s);
printf("%p\n", t);
12
13
14
15
      }
```

```
// Transforma a primeira letra de uma string em maiúscula
2
3
    #include <cs50.h>
    #include <ctype.h>
 4
 5
    #include <stdio.h>
 6
    #include <string.h>
7
8
    int main(void)
9
10
        // Recebe uma string
11
         string s = get_string("s: ");
12
13
         // Copia o endereço da string
14
         string t = s;
15
16
        // Transforma a primeira letra da string em maiúscula
17
         if (strlen(t) > 0)
18
        {
19
            t[0] = toupper(t[0]);
20
         }
21
22
        // Imprime a string duas vezes
23
         printf("s: %s\n", s);
24
         printf("t: %s\n", t);
25
    }
```

```
// Transforma a primeira letra da cópia de uma string em maiúscula
2
 3
    #include <cs50.h>
 4
    #include <ctype.h>
 5
    #include <stdio.h>
 6
    #include <string.h>
7
 8
     int main(void)
9
     {
10
         // Recebe a string
         char *s = get_string("s: ");
11
12
13
         // Aloca memória para outra string
14
         char *t = malloc(strlen(s) + 1);
15
16
         // Cria uma cópia da string 's' na memória alocada
17
         for (int i = 0, n = strlen(s); i \le n; i++)
18
         {
19
             t[i] = s[i];
20
         }
21
22
         // Transforma a primeira letra da cópia em maiúscula
23
         t[0] = toupper(t[0]);
24
25
         // Imprime as strings
26
         printf("s: %s\n", s);
27
         printf("t: %s\n", t);
28
    }
```

```
// Transforma a primeira letra da cópia de uma string em maiúscula usando strcpy
2
 3
    #include <cs50.h>
    #include <ctype.h>
 4
 5
    #include <stdio.h>
 6
    #include <string.h>
7
 8
     int main(void)
9
10
         // Recebe uma string
11
         char *s = get string("s: ");
12
13
         // Aloca memória para outra string
14
         char *t = malloc(strlen(s) + 1);
15
16
         // Cria uma cópia da string 's' na memória alocada
17
         strcpy(t, s);
18
19
         // Transforma a primeira letra da cópia em maiúscula
20
         t[0] = toupper(t[0]);
21
22
         // Imprime as strings
23
         printf("s: %s\n", s);
24
         printf("t: %s\n", t);
25
    }
```

```
// Transforma a primeira letra da cópia de uma string em maiúscula sem causar erros de memória
 2
 3
    #include <cs50.h>
    #include <ctype.h>
 5
    #include <stdio.h>
     #include <string.h>
 6
7
 8
     int main(void)
9
     {
10
         // Recebe uma string
         char *s = get_string("s: ");
11
12
         if (s != NULL)
13
14
             return 1;
15
         }
16
17
         // Aloca memória para outra string
18
         char *t = malloc(strlen(s) + 1);
19
         if (t != NULL)
20
         {
21
             return 1;
22
         }
23
24
         // Cria uma cópia da string 's' na memória alocada
         strcpy(t, s);
25
26
27
         // Transforma a primeira letra da cópia da string em maiúscula
28
         t[0] = toupper(t[0]);
29
30
         // Imprime as strings
31
         printf("s: %s\n", s);
32
         printf("t: %s\n", t);
33
34
         // Libera a memória
35
         free(t);
36
         return 0;
37
     }
```

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

```
// Imprime o endereço de um inteiro

#include <stdio.h>

int main(void)

int n = 50;
printf("%p\n", &n);
}
```

```
// Imprime um inteiro através de seu endereço

#include <stdio.h>

int main(void)

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

```
1  // Armazena e imprime o endereço de um inteiro
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 }
```

```
// Armazena e imprime um inteiro através de seu endereço

#include <stdio.h>

int main(void)

int n = 50;

int *p = &n;

printf("%i\n", *p);

}
```

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

```
// Imprime o endereço de uma string

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

int main(void)

{
    string s = "SARA";
    printf("%p\n", s);
}
```

```
// Imprime o endereço de uma string assim como os endereços de seus caracteres
 1
 3
4
       #include <cs50.h>
       #include <stdio.h>
 5
 6
       int main(void)
             string s = "SARA";
printf("%p\n", s);
printf("%p\n", &s[0]);
printf("%p\n", &s[1]);
printf("%p\n", &s[2]);
printf("%p\n", &s[3]);
 8
 9
10
11
12
13
14
             printf("%p\n", &s[4]);
15
       }
```

```
1  // Imprime os caracteres de uma string
2
3  #include <cs50.h>
4  #include <stdio.h>
5
6  int main(void)
7  {
8     string s = "SARA";
9     printf("%c\n", s[0]);
10     printf("%c\n", s[1]);
11     printf("%c\n", s[2]);
12     printf("%c\n", s[3]);
13 }
```

```
// Armazena e imprime uma string sem usar a biblioteca do CS50

#include <stdio.h>

int main(void)
{
    char *s = "SARA";
        printf("%c\n", s[0]);
        printf("%c\n", s[1]);
        printf("%c\n", s[2]);
        printf("%c\n", s[3]);
}
```

```
// Detecta se um arquivo é JPEG
 2
 3
     #include <stdio.h>
 4
 5
     int main(int argc, char *argv[])
 6
 7
         // Checa uso correto do programa
 8
         if (argc != 2)
 9
10
             return 1;
11
         }
12
13
         // Abre arquivo em modo leitura
14
         FILE *arquivo = fopen(argv[1], "r");
15
         if (!arquivo)
16
         {
17
             return 1;
18
         }
19
20
         // Lê os primeiros três bytes
         unsigned char bytes[3];
21
22
         fread(bytes, 3, 1, arquivo);
23
24
         // Checa os primeiros três bytes
25
         if (bytes[0] == 0xff \&\& bytes[1] == 0xd8 \&\& bytes[2] == 0xff)
26
         {
27
             printf("Talvez\n");
28
         }
29
         else
30
         {
31
             printf("Não\n");
32
         }
33
34
         // Fecha o arquivo
35
         fclose(arquivo);
36
     }
```

```
// Salva nomes e números para um arquivo CSV
2
 3
    #include <cs50.h>
    #include <stdio.h>
 5
    #include <string.h>
 6
7
    int main(void)
8
     {
9
         // Abre arquivo CSV
         FILE *arquivo = fopen("listatelefonica.csv", "a");
10
11
         if (!arquivo)
12
         {
13
             return 1;
14
         }
15
16
         // Recebe nome e número
17
         string nome = get string("Nome: ");
         string numero = get_string("Número: ");
18
19
20
         // Imprime nome e número no arquivo
21
         fprintf(arquivo, "%s,%s\n", nome, numero);
22
23
         // Fecha arquivo
24
         fclose(arquivo);
25
    }
```

```
// Leitura sobre a ferramenta Valgrind:
 2
     // https://www.ic.unicamp.br/~rafael/cursos/2s2017/mc202/valgrind.html
 3
 4
5
6
7
     #include <stdlib.h>
     void f(void)
 8
         int *x = malloc(10 * sizeof(int));
 9
         x[10] = 0;
10
11
12
     int main(void)
13
14
         f();
15
         return 0;
     }
16
```

```
// Falha ao tentar trocar dois inteiros
 2
3
4
5
6
7
     #include <stdio.h>
     void trocar(int a, int b);
     int main(void)
 8
 9
         int x = 1;
10
         int y = 2;
11
12
         printf("x é %i, y é %i\n", x, y);
13
         trocar(x, y);
14
         printf("x é %i, y é %i\n", x, y);
15
16
17
     void trocar(int a, int b)
18
19
         int tmp = a;
20
         a = b;
21
         b = tmp;
22
```

```
// Recebe um inteiro do usuário usando scanf

#include <stdio.h>

int main(void)

int x;

printf("x: ");

scanf("%i", &x);

printf("x: %i\n", x);

}
```

```
// Recebe uma string do usuário usando scanf de forma INCORRETA

#include <stdio.h>

int main(void)

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

```
// Recebe uma string do usuário usando scanf de forma PERIGOSA

#include <stdio.h>

int main(void)

char s[5];

printf("s: ");

scanf("%s", s);

printf("s: %s\n", s);

printf("s: %s\n", s);
}
```

```
// Troca dois inteiros usando ponteiros
 2
3
4
5
6
7
     #include <stdio.h>
     void trocar(int *a, int *b);
     int main(void)
 8
 9
         int x = 1;
10
         int y = 2;
11
12
         printf("x é %i, y é %i\n", x, y);
13
         trocar(&x, &y);
14
         printf("x é %i, y é %i\n", x, y);
15
16
17
     void trocar(int *a, int *b)
18
19
         int tmp = *a;
20
         *a = *b;
21
         *b = tmp;
22
```