Nathan Hamilton
CSCD 437-01
Homework 4

a.

```
// Ensure passed in arg is the correct size, and the copied string ends with \0
int foo(char *arg, char *out)   // Add parameter size_t n for max string size
{
    strcpy(out, arg);   // Change to strncpy(out, arg, n);
    // Add a check for if n is max size (64 in this case), change out[63] = '\0';
    return 0;
}
int main(int argc, char *argv[])
{
    char buf[64];
    if (argc != 2)
    {
        fprintf(stderr, "a: argc != 2\n");
        exit(EXIT_FAILURE);
    }
    // Add variable size_t len = strlen(argv[1]); for the length of the string passed in
    // Add a check for if len > 64 then foo(argv[1], buf, 63);
    // Else foo(argv[1], buf, len);
    foo(argv[1], buf);
    return 0;
}
```

b.

```
// Ensure length variables are of type size_t and space is left for '\0'
int foo(char *arg)
{
    char buf[128];
    int len, i;     // Change int to size_t

    len = strlen(arg);
    if (len > 136)     // Change 136 to 128
        len = 136;     // Change 136 to 128

    for (i = 0; i <= len; i++)
        buf[i] = arg[i];
    return 0;
}
int main(int argc, char *argv[])
{
    if (argc != 2)
    {
        fprintf(stderr, "b: argc != 2\n");
        exit(EXIT_FAILURE);
    }
    foo(argv[1]);
    return 0;
}
```

c.

```
// Ensure length variables are of size_t and buf does not overrun allotted size
int bar(char *arg, char *targ, int ltarg)     // Change Itarg type to size_t
{
    int len, i;     // Change int to size_t
    len = strlen(arg);
    if (len > ltarg)
        len = ltarg;
    for (i = 0; i <= len; i++)
        // Add check if len is 128 then targ[127] = '\0'
        targ[i] = arg[i];
    return 0;
}
int foo(char *arg)
{
    char buf[128];
    bar(arg, buf, 140);     // Change 140 to 128
    return 0;
}
int main(int argc, char *argv[])
{
    if (argc != 2)
    {
        fprintf(stderr, "c: argc != 2\n");
        exit(EXIT_FAILURE);
    }
    foo(argv[1]);
    return 0;
}
```

d.

```
// Ensure length veriables are of type size_t; use arglen instead of calling strlen
int foo(char *arg, short arglen)     // Change short to size_t
{
    char buf[1024];
    int i, maxlen = 1024;     // Change int to size_t
    If (arglen < maxlen)
    {
        for (i = 0; i < strlen(arg); i++)     // Change strlen(arg) to arglen
        buf[i] = arg[i];
    }
    return 0;
}
int main(int argc, char *argv[])
{
    if (argc != 2)
    {
        fprintf(stderr, "d: argc != 2\n");
        exit(EXIT_FAILURE);
    }
    foo(argv[1], strlen(argv[1]));
    return 0;
}
```