HW exercise 1 – Dry Part

Original function:

```
char* stringDuplicator(char* s, int times) {
    assert(!s); //bug (1)
    assert(times > 0);
    int LEN = strlen(s); //convention (1)
    char* out = malloc(LEN * times); // bug (2+3), convention (2)
    assert(out);
    for (int i = 0; i <= times; i++) {//bug (4)
        out = out + LEN; //bug (6)
        strcpy(out, s);// bug (5)
    }
    return out; //bug (6)
}
```

Code bugs:

- (1)- we want to check if char* s is null but if we do (!s) it will give us true if the string is null and won't stop the run
- (2)- it should be (len * times + 1) for the null sign at the end of the string
- (3)- should be sizeofchar in the brackets --> (len * times * sizeof(char))
- (4)- it counts one excess time and reaching unwanted memory --> (i < times) and not (i <= times)
- (5)- the strcpy should be before we promote the string to the place we want to copy the string
- (6)- when you move forward you lose the pointer to the begining of the string so we will return a pointer to the begining of the last small string.

Code conventions bugs:

- (1)- variables names shoudln't be in capital letters
- (2)- should be a type before the malloc comand. for example:
  (char*) malloc(expression)

Fixed function:

```
char* stringDuplicator(char* s, int times) {
    assert(s);
    assert(times > 0);
    int len = strlen(s);
    char* out = (char*)malloc(((len * times) + 1)*sizeof(char));
    assert(out);
    char* temp = out;
    for (int i = 0; i < times; i++) {
        strcpy_s(temp, (len + 1) *sizeof(char), s);
        temp = temp + len;
    }
    return out;
}
```