תרגיל בית 1 – חלק יבש

2.1.1 סעיף א':

שגיאות תכנות:

1. אין בדיקה האם המחרוזת s שמקבלים מהמשתמש היא NULL.
assert(!s) לא יעצור את התכנית אם s = NULL אלא יעצור אותה אם s שונה מ- NULL ולכן זו שגיאה.

2. לאחר הקצאת הזיכרון בשורה char* out = malloc(LEN*times);
אין בדיקה מפורשת לכך ש malloc() הצליחה. אמנם assert(out) יעצור את התכנית אם הקצאת הזיכרון תיכשל, אך אם נגדיר #define NDEBUG בתחילת התכנית אז assert(out) יעלם ולכן צריך לבדוק באופן מפורש.

3. בשורה char* out = malloc(LEN*times); צריך להקצות עוד בית אחד על מנת לשים '\0' בסוף המחרוזת.

4. לולאת ה-for מבצעת times+1 איטרציות, ולכן נקבל שבאיטרציה האחרונה strcpy(out,s) תעתיק את המחרוזת לזיכרון שלא הקצנו ותגרור segmentation fault.

5. בלולאת ה-for, מכיוון שקודם כל מקדמים את המצביע, ואז מעתיקים את המחרוזת למקום המתאים, נקבל שLEN הבתים הראשונים בout יכילו ערכי זבל כי לא איתחלנו אותם.

6. בסוף הפונקציה, מוחזר מצביע לסוף המחרוזת ולא לתחילת המחרוזת.

שגיאות קונבנציה:

1. הסוגר הפותח של בלוק הפונקציה לא מופיע בשורה נפרדת.

2. המשתנה המקומי בשם LEN צריך להיות באותיות קטנות. כמו למשל "len".

3. שם המצביע s אינו ברור. הוא צריך להיות "string" או "str".

4. אין שימוש מתאים בהזחות (identation) בשורות הקוד שנמצאות בתוך לולאת ה- for.
כלומר "צריכים להופיע" רווחים לפני הפקודות שבתוך הלולאה.

2.1.2 סעיף ב':

```
char* stringDuplicator(char* str, int times)
{
    assert(str != NULL);
    assert(times > 0);

    int len = strlen(str);
    char* out = malloc(sizeof(char)*(len*times+1));
    if (out == NULL)
    {
      printf("Memory error\n");
      return NULL;
    }

    char* result = out;
    for (int i=0; i < times; i++)
    {
       strcpy(out,str);
       out += len;
    }
    return result;
}
```