שאלה 1

```
ErrorCode mergeSortedLists(Node list1, Node list2, Node *merged_out)
{

    if(list1 == NULL || list2 == NULL || merged_out == NULL)
    {
        return NULL_ARGUMENT;
    }

    int list1Length = getListLength(list1);
    int list2Length = getListLength(list2);

    if(list1Length == 0 || list2Length == 0)
    {
        return EMPTY_LIST;
    }

    if(!isListSorted(list1) || !isListSorted(list2))
    {
        return UNSORTED_LIST;
    }

    Node list1Pointer = list1;
    Node list2Pointer = list2;

    Node *mergedPointer = merged_out;
    while(list1Pointer != NULL || list2Pointer != NULL)
    {
        *mergedPointer = nodeCreate(0);
        if(*mergedPointer == NULL)
        {
            // allocation failed - destroy list and exit
            listDestroy(*merged_out);
            merged_out = NULL;
            return MEMORY_ERROR;
        }

        if(list1Pointer!=NULL && (list2Pointer == NULL || list1Pointer-
>x <= list2Pointer->x) )
        {
            (*mergedPointer)->x = list1Pointer->x;
            list1Pointer = list1Pointer->next;
        }
        else
        {
            (*mergedPointer)->x = list2Pointer->x;
            list2Pointer = list2Pointer->next;
```

```
        }
        mergedPointer = &(*mergedPointer)->next;
    }


    return SUCCESS;

}
```

שאלה 2

שגיאות תכנות

1. הassert בהתחלה מוודא שהמחרוזת שנקלטת היא NULL, במקום לוודא שהיא לא כזו.
2. המקום בזיכרון שמקצים ל out לא כולל את התווים '0/', שיועתקו באמצעות strcpy עבור כל עותק של s.
3. בתוך הלולאה, out מתקדם לפני שהמחרוזת מועתקת אליו, ולכן העותק הראשון יוכנס החל ממיקום LEN, והאיטרציה האחרונה תנסה לגשת למקום בזיכרון שלא הוקצה לout.
4. הפונקציה מחזירה את המשתנה out, אבל לאחר ביצוע הלולאה הוא מצביע רק לעותק האחרון של s בתוך המחרוזת, ולא לתחילת המחרוזת החדשה.
5. הפונקציה לא תחזיר NULL אם יש שגיאה.

שגיאות קונבנציה

1. שם המשתנה s אינו ברור ואינו מעיד על משמעות המשתנה.
2. הגדרת המשתנה LEN באותיות גדולות.
3. אין הזחה בגוף הפונקציה או בתוך הלולאה.
4. שם הפונקציה לא מנוסח כפועל.

פונקציה מתוקנת

```
#include <stdlib.h>

#include <string.h>

#include <assert.h>


char *duplicateString(char *str, int times) {

   assert(str);

   assert(times > 0);

   int len = strlen(str);

   char *out = malloc((len+1) * times);

   if(!out){

      return NULL;

   }

   char *iterator = out;

   for (int i = 0; i < times; i++) {

     strcpy(iterator, str);

     iterator = iterator + len;

   }

   return out;

}
```

שאלה 2

שגיאות תכנות

6. הassert בהתחלה מוודא שהמחרוזת שנקלטת היא NULL, במקום לוודא שהיא לא כזו.
7. המקום בזיכרון שמקצים ל out לא כולל את התווים '0/', שיועתקו באמצעות strcpy עבור כל עותק של s.
8. בתוך הלולאה, out מתקדם לפני שהמחרוזת מועתקת אליו, ולכן העותק הראשון יוכנס החל ממיקום LEN, והאיטרציה האחרונה תנסה לגשת למקום בזיכרון שלא הוקצה לout.
9. הפונקציה מחזירה את המשתנה out, אבל לאחר ביצוע הלולאה הוא מצביע רק לעותק האחרון של s בתוך המחרוזת, ולא לתחילת המחרוזת החדשה.
10. הפונקציה לא תחזיר NULL אם יש שגיאה.

שגיאות קונבנציה

5. שם המשתנה s אינו ברור ואינו מעיד על משמעות המשתנה.
6. הגדרת המשתנה LEN באותיות גדולות.
7. אין הזחה בגוף הפונקציה או בתוך הלולאה.
8. שם הפונקציה לא מנוסח כפועל.

פונקציה מתוקנת

```
#include <stdlib.h>

#include <string.h>

#include <assert.h>


char *duplicateString(char *str, int times) {

   assert(str);

   assert(times > 0);

   int len = strlen(str);

   char *out = malloc((len+1) * times);

   if(!out){

      return NULL;

   }

   char *iterator = out;

   for (int i = 0; i < times; i++) {
```

```
    strcpy(iterator, str);
    iterator = iterator + len;
  }
  return out;
}
```