שאלה 2 (25 נקודות)

סעיף א

נתונות שתי מחרוזות num1, num2 (המחרוזות מסתיימות ב-'\0') המייצגות שני מספרים חיוביים הקסדצימליים (בבסיס 16). עליך לממש פונקצית השוואה ביניהן שמקבלת כקלט את num1 ואת num2 ומחזירה 1 אם num1 > num2, 1- אם num1 < num2 ו-0 במקרה של שוויון.

אין להניח שאורך המחרוזות חסום, כלומר המספרים ההקסדצימליים יכולים להיות גדולים מהמספרים הגדולים ביותר הניתנים לייצוג ב- int ועל-כן פתרון המתרגם את המחרוזות למספר לא יתקבל. אין צורך לבדוק את תקינות הקלט (כלומר ניתן להניח שהמחרוזות אכן מייצגות מספר הקסדצימלי). כמו כן ניתן להניח שהאותיות האנגליות במחרוזות גדולות, וכן שהמספרים אינם מכילים אפסים לפני המספר עצמו (למשל: 00FFF)

לדוגמא :

FFF < 123A < 123B < 11D11

ניתן להשתמש בפונקציות הספרייה הבאות:

- `int isalpha(int ch)` – מחזירה ערך שונה מ-0 אם ch הוא אות (קטנה או גדולה), או 0 אחרת.
- `int isdigit(int ch)` – מחזירה ערך שונה מ-0 אם ch הוא ספרה בין 0 ל-9, או 0 אחרת.
- `size_t strlen(char* str)` – מחזירה את אורך המחרוזת str.

```
int hex_cmp(char* num1, char* num2)
{
  if(strlen(num1) > strlen(num2))
    return 1;
  else if(strlen(num1) < strlen(num2))
    return -1;
  for(;*num1; num1++, num2++) {
    int i1 = isalpha(*num1)? (*num1-'A'+10) : (*num1-'0');
    int i2 = isalpha(*num2)? (*num2-'A'+10) : (*num2-'0');

    if(i1 == i2)
      continue;
    return((i1 > i2)? 1 : -1);
  }

    return 0;
}
```

סעיף ב

יש לממש פונקציה המקבלת מערך ממוין של מחרוזות המייצגות מספרים הקסדצימליים (מהקטן לגדול), את גודלו n, ואת המחרוזת num. על הפונקציה להחזיר את מספר המופעים של num במערך (אם המחרוזת לא נמצאת במערך יש להחזיר 0).
ניתן (ואף מומלץ) לממש פונקציות עזר וכן להשתמש בפונקציות מסעיפים קודמים.
יש לממש את הפונקציה בסיבוכיות הזמן הטובה ביותר האפשרית, ובסיבוכיות המקום הטובה ביותר עבור סיבוכיות זמן זו. מהן סיבוכיות המקום והזמן כפונקציה של n ושל K בהנחה שאורך המחרוזת הארוכה ביותר אינו גדול מ-K?

```
int count_appearances(char* nums[], int n, char* num)
{
    int high=n-1, low=0, mid=n/2;
    int mid1, mid2;
    while(low < high-1) {
        if(hex_cmp(nums[mid], num) >= 0)
            high = mid;
        else
            low = mid+1;
        mid = (low+high)/2;
    }
    mid1 = (hex_cmp(nums[low], num) == 0)? low : high;
    if(hex_cmp(nums[mid1], num) != 0)
        return 0;
    high=n-1, low=0, mid=n/2;
    while(low < high-1) {
        if(hex_cmp(nums[mid], num) <= 0)
            low = mid;
        else
            high = mid-1;
        mid = (low+high)/2;
    }
    mid2 = (hex_cmp(nums[low], num) == 0)? low : high;
    if(hex_cmp(nums[mid2], num) != 0)
        return 0;

    return (mid2-mid1+1);
}
```

סיבוכיות זמן: חיפוש התחלת וסוף הרצף נעשים עפ"י חיפוש במערך ממויין, מה שנותן O(logn). בכל פעם קוראים ל-hex cmp() שעובד בסיבוכיות O(k), כאשר שאר הפעולות הן מסד"ג O(1). לכן סיבוכיות הזמן היא O(klogn).

סיבוכיות מקום נוסף: O(1).