שאלה 2 (25 נקודות)

נתון מערך order באורך קבוע של 26 (שימו לב להערה בהמשך). מערך זה מכיל את כל האותיות הקטנות בא"ב האנגלי, כל אות פעם אחת בדיוק. המערך מגדיר סדר חדש על אותיות הא"ב, כאשר על פי סדר זה, אות אחת תחשב לקטנה מאות אחרת אם ורק אם אות זו נמצאת במערך order לפני האות האחרת.

לדוגמה, עבור המערך order הבא מתקיים b<a, כיוון ש- b מופיע לפני a במערך. לעומת זאת, יתר האותיות במקרה זה הן בסדר לכסיקוגרפי רגיל.

order[26] = { 'b','a','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z'} ;

הערה: בשאלה זו, מספר האותיות בא"ב האנגלי (26) יכול להיחשב כקבוע לצורך חישובי סיבוכיות.

סעיף א

השלימו את הפונקציה הבאה, המקבלת כקלט מחרוזת s המכילה אותיות אנגליות קטנות בלבד, את אורכה של המחרוזת n (לא כולל תו ה-null), ומערך order המתאר סדר של אותיות הא"ב. הפונקציה מחזירה 1 אם האותיות ב-s מסודרות על פי הסדר המוגדר במערך order, ו-0 אחרת (עבור מחרוזת ריקה החזירו 1). **יש לממש את הפונקציה בסיבוכיות זמן ומקום טובים ככל הניתן**. פתרון לא אופטימאלי יזכה בניקוד מופחת. כמו כן, השלימו את סיבוכיות האלגוריתם שלכם במקום המתאים למטה, כפונקציה של n.

```
int is_ordered(char *s, int n, char order[26]) {

  int i=0, j=0;

  while (j<n) {
    while (i<26 && order[i] != s[j])
      i++;
    if (i==26)
      return 0;
    j++;
  }
  return 1;
}
```

הערה: פתרון נפוץ אחר הינו להגדיר פונקצית עזר compare(char a, char b, char order[26]) המשווה בין שני תווים a ו-b בהתאם לסדר המוגדר ב-order, ובאמצעות פונקציה זו לבצע סריקה של המחרוזת s ולוודא שכל שני תווים עוקבים בה מסודרים נכון.

סיבוכיות זמן: θ (n) סיבוכיות מקום נוסף: θ (1)

<u>סעיף ב</u>

בסעיף זה נממש פונקציה המבצעת חיפוש של אות נתונה במחרוזת ממוינת. השלימו את הפונקציה הבאה, המקבלת מחרוזת s הממוינת על פי סדר המוגדר ב-order, את אורך המחרוזת n (לא כולל תו ה-null) וכן אות לחיפוש x. על הפונקציה להחזיר את האינדקס של **המופע האחרון** של x במחרוזת s, או 1- אם x אינה מופיעה המחרוזת.

לדוגמה, עבור הסדר order בדף הקודם והמחרוזת הבאה (n=9):
char s[] = "bbacdddef"

על הפונקציה להחזיר 2 במקרה של חיפוש האות a, 1 במקרה של חיפוש האות b, 6 במקרה של חיפוש האות d ו-(1-) במקרה של חיפוש האות y.

יש לממש את הפונקציה בסיבוכיות זמן ומקום טובים ככל הניתן. פתרון לא אופטימאלי יזכה בניקוד מופחת. כתבו את סיבוכיות האלגוריתם שלכם במקום המתאים:

סיבוכיות זמן:	θ (log n)	סיבוכיות מקום נוסף:	θ (1)

```
int search(char *s, int n, char order[26], char x) {

  int low=0, high=n-1, mid, i;
  char char_table[26];      // char_table[i] contains the index
                            // of the letter 'a'+i in order[]
  for (i=0; i<26; ++i)
    char_table[order[i] - 'a'] = i;

  while (low <= high) {
    int mid = (high+low)/2;
    int compare = char_table[x-'a'] - char_table[s[mid]-'a'];

    // if characters are equivalent and it is the last occurrence
    if (compare == 0 && (mid==n-1 || s[mid+1]!=s[mid]))
      return mid;
    else if (compare < 0)
      high = mid-1;
    else
      low = mid+1;
  }
  return -1;
}
```