מבני נתונים (מצטיינים) - סמסטר ב' תשפ"א

מטלה 2

הנחיות:

- מטלה זו הינה להגשה ביחידים. אין למסור ולקבל פתרון מתלמיד אחר העתקה תגרור **לפסילה מלאה** של המטלה למעתיק והמועתק.
- בשאלות 1,2 יש להקליד את התשובה בקובץ טקסט או וורד (וכו') בשם Ex2. בשאלות 3,4 יש לעדכן את הפונקציות בקובץ הנתון Ex2.java.
- יש להגיש את שני קבצי המטלה בקובץ ZIP (ולא כל דחיסה אחרת). שם קובץ ה- ZIP יהיה מספר ת.ז. של התלמיד בלבד (ולא כל שם אחר). אין להגיש קבצים או תיקיות מיותרים. סטייה מהנחיות אלו תגרור הורדה בציון.

שאלה 1:

בכיתה ראינו מימוש של אלגוריתם מיון מהיר. להלן מימוש רקורסיבי של מיון מהיר (לא כולל מימוש partition)

```
static void QuickSort(int arr[], int low, int high)
  {
    if (low < high) {
      int p = partition(arr, low, high);
      QuickSort (arr, low, p - 1);
      QuickSort (arr, p + 1, high);
    }
  }
```

בשאלה זו, ננסה להבין האם ניתן לחסוך בקריאות רקורסיביות במיון מהיר. בכל הסעיפים אנו מניחים מערך קלט בגודל n איברים.

1. נניח כי partition בוחרת את איבר הציר עפ"י האיבר הראשון במערך. מהו סדר הגודל של עומק הרקורסיה המתקבל במקרה הטוב ביותר? מהו סדר הגודל של עומק הרקורסיה במקרה הגרוע? הסבירו ותנו דוגמא לקלט שיוביל לעומק רקורסיה מינימלי ומקסימלי.

2. להלן מימוש שונה של פונקצית QuickSort:

```
void QuickSort(int arr[], int low, int high)
{
  while (low < high)
  {
    int p = partition(arr, low, high);
    if (p - low < high - p)
    {
      QuickSort(arr, low, p - 1);
      low = p + 1;
    }
    else
    {
```

```
        QuickSort(arr, p + 1, high);
        high = p - 1;
    }
  }
}
```

מהו סדר הגודל של עומק הרקורסיה המתקבל במקרה הגרוע? הסבירו.

שאלה 2:

בשאלה זו נניח מערך קלט המורכב מפרמוטציה של המספרים השלמים 1 עד n, כאשר n שייך לקבוצת המספרים הטבעיים. לכן יש n! מערכים שונים שניתן לקבל כקלט.

יוסי וענת מציעים שני מימושים שונים לפונקצית partition:

```
static int partition_yossi(int[] arr, int low, int high){
int pivot = arr[low];
int i = low-1 , j = high + 1;
while (true) {
        do {
                i++;
        } while (i < arr.length && arr[i] <= pivot);
        do {
                j--;
        } while (arr[j] > pivot);
        if (i >= j) {
                int temp = arr[j];
                arr[j] = pivot;
                arr[low] = temp;
                return j;
        }
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
        }
}

static int partition_anat(int [] arr, int low, int high){
int pivot = arr[high];
int i = (low - 1);
for (int j = low; j <= high- 1; j++){
        if (arr[j] <= pivot){
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
        }
}
int temp = arr[i+1];
arr[i+1] = arr[high];
arr[high] = temp;
return (i + 1);
}
```

1. מהו **ממוצע** מספר החילופים (קריאות לפונקצית Swap) בגירסא של ענת כאשר הממוצע הוא כל הקלטים האפשריים (קלט = מערך הנובע מפרמוטציה ספציפית)?
2. איזה מהמימושים של partition יוביל למספר קטן יותר של חילופים ב**ממוצע** על כל הקלטים האפשריים?
3. בהנתן קלט שהינו מערך שכולו אפסים, מה יהיה זמן הריצה של מיון מהיר אשר משתמש ב partition של ענת לעומת הpartition של יוסי?

שאלה 3:

1. בכיתה ראינו מימוש רקורסיבי של מיון מיזוג. כתבו מימוש איטרטיבי (ללא קריאות רקורסיביות) של מיון מיזוג.

```
public static void mergeSort2(int[] arr)
```

2. בכיתה ראינו מימוש של פונקצית merge הממזגת שני מערכים ממוינים. כזכור, merge השתמשה בזיכרון נוסף בגודל O(n) כאשר n הינו מספר האיברים במערך התוצאה (לאחר המיזוג). כתבו מימוש חדש של merge:

```
public static void merge(int[] arr, int low, int mid, int high, int max_num)
```

אשר מבצעת מיזוג של החלקים הממוינים [low,mid] ו-[mid+1,high] ללא שימוש בזיכרון נוסף התלוי בגודל הקלט (ניתן להשתמש ב-O(1) זכרון בלבד).

הנחות: המערך הוא של שלמים. ערכי האיברים במערך קטנים ממש משורש של ערך המקסימלי השלם הנתמך במכונה. ניתן להשתמש במשתנה העזר max_num כרצונכם (למשל, ניתן להחליט לאחסן בו מספר שלם כלשהו).

רמז: תחת ההנחה שערך האיברים קטנים ממש משורש של ערך המקסימלי (לדוגמה, ערך int בג'אוה) ניתן ליצג שני מספרים שונים בתוך int אחד: אם x>y וגם x>z אזי מתקיים z = (z+y*x) % x וגם y = (z+y*x) / x.

3. בהינתן מערך המכיל M רשימות מקושרות ממוינות (של מספרים שלמים), כל אחת באורך N, מזגו אותן לרשימה מקושרת אחת ממוינת. סיבוכיות הפתרון הנדרשת (O(NMlog(M)).

```
public static LinkedList<Integer> join(LinkedList<Integer>[] arr)
```

שאלה 4:

נתון מערך לא ממוין עם n איברים (מספרים שלמים), כאשר כל המספרים הם בין 0 ל-n. כתבו פונקציה המחזירה את ההפרש המקסימלי בין שני איברים סמוכים במערך בצורתו הממוינת (אילו היה ממוין). סיבוכיות הפתרון הנדרשת O(n).

```
public static int diff(int[] arr)
```

ב ה צ ל ח ה !