מבוא למדעי המחשב – כתה יא-1

תאריך: 22.1.2026

תורים דף עבודה #2 – תרגילים (מתקדם)

ודגשים על: כתיבת אלגוריתם + חלוקה לתת-משימות

מטרות:

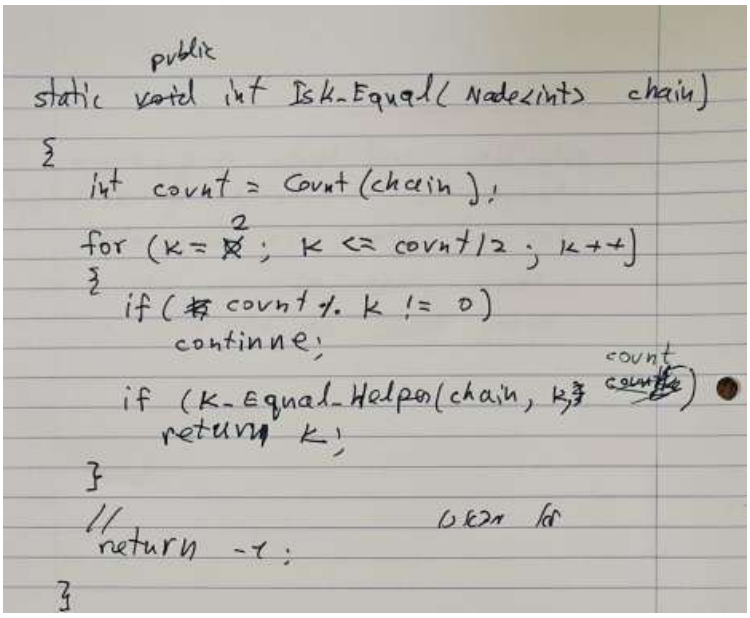
1. עבודה מתקדמת עם תורים – שאלות מורכבות ברמה דומה לבגרות
2. אימון בכתיבת אלגוריתם וחלוקת משימה לתת-משימות

משימה #1

עיין בשאלה #1 בבחינה האחרונה (פיצול רשימה לתת רשימות, והחזרת מערך של מצביעים לתת רשימות האלו).

א. כתבו את האלגוריתם לסעיף א' של השאלה. (דגש: הרמה הגבוהה. אין צורך לפרט "איך עושים" אלא "מה עושים". הקפידו על מספרי צעדים ושימוש בפסאודו קוד (ראו דוגמא במשימה הבאה).

ב. נתון הפתרון שאני כתבתי לשאלה הזאת – רק הפונקציה העיקרית שמבוקשת בשאלה. עיינו בפתרון שלי וחשבו עליו בצורה ביקורתית. האם זה קשה? קל? האם זה עוזר? (אגב, העיקרון הוא מה שחשוב – אין לי ציפייה שאתם תכתבו קוד בדיוק כמו מה שאני כתבתי – אבל כן להקפיד על הרמה העליונה וקריאה לתתי פונקציות).

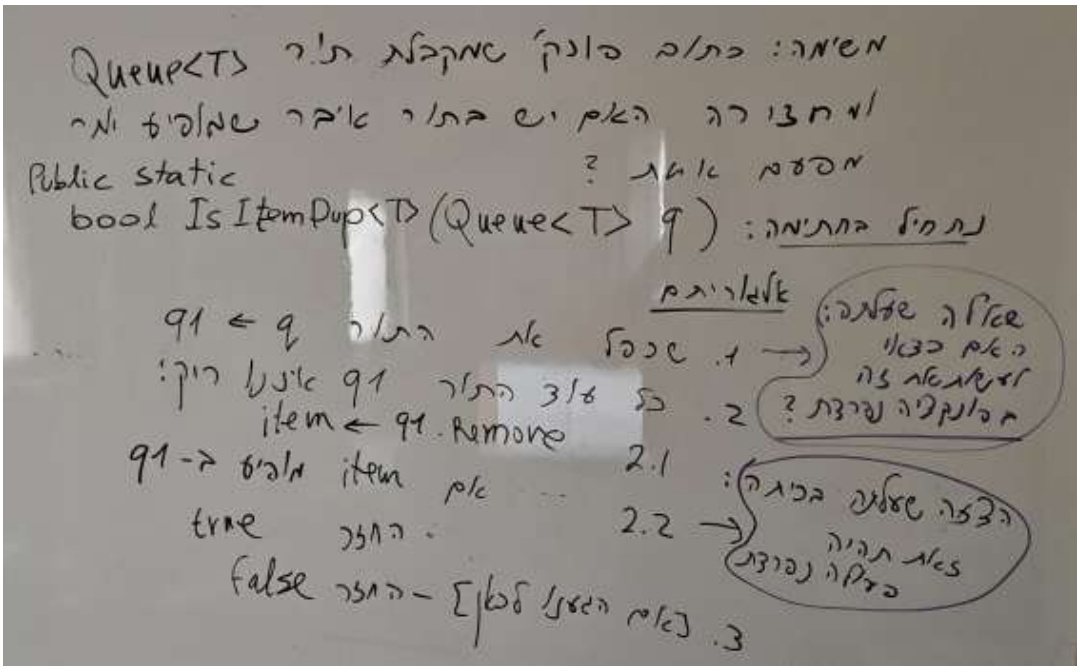


```
public
static void int Isk_Equal( NodeInts chain)
{
   int count = Count(chain);
   for (k = 2; k <= count/2; k++)
   {
      if (count % k != 0)
         continue;
      if (k_Equal_Helper(chain, k, count))
         return k;
   }
   //
   return -1;
}
```

משימה #2

ג. נתונה השאלה שסיימנו איתה את השיעור האחרון – צילום מסך של הלוח (שחזרתי מהזיכרון). מה שעליכם לעשות:

1. לחשוב על שתי השאלות שמוקפות בעיגול על הלוח. השאלה הראשונה: האם צריך לשכפל תור? השאלה השנייה: האם משכפלים פעם אחת או בכל איטרציה?
2. לכתוב את המימוש, בכתב יד, על דף נייר או במחברת - אבל רק של הרמה העליונה (בדומה למה שאתם רואים במשימה הקודמת).
3. אין (אסור!) לכתוב את הקוד של הפונקציות שקוראים להן. אבל כן צריך לכתוב את החתימה שלהן (שם פונקציה + ארגומנטים שהיא מקבלת + הטיפוס שהיא מחזירה).



חלק ב': משימות תירגול והבנה של תורים - שאלות מדורגות עד לרמה דומה לבגרות

השלימו את כל התרגילים האלו בבית – גם את אלו שנפתרו בכיתה – חשוב שתפתרו בכוחות עצמכם.

לכל תרגיל – כחלק מתהליך החשיבה – כתבו אלגוריתם בפסאודו קוד. לאחר מכן, כתבו את הפונקציה המבוקשת. ברוב המקרים יש צורך לפרק לתת משימות

בסעיף או בתרגיל מסוים – מותר ורצוי להשתמש בפונקציה מסעיף קודם.

הנחיות לכל המשימות:

- בכל התרגילים עליכם להשתמש רק בתור, כלומר במחלקה `Queue<T>` !
- מותר, וברוב המקרים זה הכרחי, להשתמש בתור עזר.
- כתיבת אלגוריתם – (1) זה נדרש לבגרות (2) לאחר מכן תרגום לקוד נעשה קל יותר.
- חלוקה לתת משימות – (1) זה נדרש. (2) זה מקל על הכתיבה. (3) מאפשר שיתוף קוד בין שאלות.

תהליך הפתרון והתירגול

1. לגבי תרגילים שפתרנו בכיתה – קראו את הפתרון בבית והבינו אותו. לאחר מכן, סגרו את המחברת וכתבו אותו בעצמכם.
2. תרגילים אחרים:
   א. השתמשו בציור להבין את תהליך הפתרון.
   ב. תרגמו את הפתרון לסדרת צעדים (אלגוריתם).
   ג. זהו תת משימות (פונקציות).
   ד. כיתבו את הפתרון בכתב יד (בדומה לבחינה).
3. בסיום – כתבו את הפתרונות שלכם ב `Visual Studio` באמצעות הפרויקט בדף עבודה מספר 1# (הקבצים ב `GDRIVE` : בתיקיה בקישור הזה – מכילים את את המשימות של התרגול הקודם. הם מכילים התייחסות לתרגילים באוסף הזה (כשתריצו את התכנית ב- VS תראו אפשרות שנקראת " `Advanced-Bagrut level menu`" והפונקציות שאתם צריכים לכתוב לחלק הזה נמצאות בקובץ ("`Program_Advanced.cs`").

<u>תרגילים עם תור עדיפויות (של מספרים שלמים.)</u>

<u>תרגיל #1</u> <u>הכנסת איבר לתור מסודר.</u>

נתון: תור של מספרים שלמים, המסודר בסדר יורד (כלומר: המספר הגדול ביותר נמצא בראש התור).

נדרש: כתבו פונקציה שמקבלת תור כזה ומספר חדש X. הפונקציה מכניסה את X למקום המתאים כך שהתור יישאר מסודר בסיום הפעולה.

חתימה הפונקציה המוצעת:

```
static void PriorityInsert(Queue<int> q, int x)
```

<u>תרגיל #2</u> <u>הפיכת תור של שלמים לתור מסודר.</u>

נתון: תור של מספרים שלמים, ללא סדר כלשהו.

נדרש: כתבו פונקציה שמארגנת את התור כך שיהיה מסודר בסדר יורד.

חתימה מוצעת:

```
static void MakePriorityQueue(Queue<int> q)
```

תרגילים עם תור עדיפויות (של מחלקה)

תרגילים עם המחלקה Student

בתרגילים הבאים נרצה להשתמש בתור שמכיל מחלקה. לשם כך נגדיר את המחלקה Student :

```
public class Student
{
    private string name;
    private int grade;

    public Student(string name, int grade)
    {
        this.name = name;
        this.grade = grade;
    }

    public string GetName()  { return name; }
    public int GetGrade()  { return grade; }
    public void SetName(string name) { this.name = name; }
    public void SetGrade(int grade) { this.grade = grade; }

    // for printing or debugging
    public override string ToString()
    {
        return (name + "  grade: " + grade);
    }
}
```

תרגיל #3 הכנסת תלמיד לתור עדיפויות של תלמידים

בבית הספר פועל מערך תיגבור פרטני. התלמידים (כל תלמיד מיוצג על ידי אובייקט מהמחלקה Student) נכנסים לתור לקבלת עזרה. העדיפות נקבעת כך:

- תלמיד עם ציון נמוך יותר יקבל עדיפות גבוהה יותר. כלומר: **ציון נמוך יותר = עדיפות גבוהה יותר**.
- אם לשני תלמידים יש אותו ציון – נשמר סדר ההגעה לתור

נתון תור של תלמידים המסודר כבר לפי כללי העדיפות. כתוב פונקציה סטטית ששמה הוא PriorityInsert, שמקבלת תור כזה ותלמיד חדש, ומכניסה את התלמיד למקום המתאים כך שהתור יישאר מסודר.

תרגיל #4 ארגון תור לא מסודר שיהיה תור עדיפויות

נתון תור של תלמידים **שאינו** מסודר לפי עדיפות. כתבי פונקציה ששמה הוא MakePriorityQueue., הפונקציה מקבלת תור לא מסודר, ומשנה אותו להיות תור לפי כלל לפי כללי העדיפות שתוארו בשאלה הקודמת.

אם לא נמצא תלמיד עם השם המתאים, אז התור נשאר ללא שינוי.

תרגיל #5 עדכון ציון של תלמיד ושינוי מיקומו בתור

תלמיד נכנס לתור לתיגבור. אחרי מבחן קצר מעדכנים לו ציון, ולכן ייתכן שהוא צריך לשנות מקום בתור. העדיפות מוגדרת כמו קודם. יש לכתוב פונקציה בשם UpdateGradeAndReorder שמקבלת תור מסודר, שם תלמיד, וציון חדש.

א. כתבו אלגוריתם לפונקציה הדרושה.
ב. זהו את הפונקציות שנכתבו עד כה ושאפשר להשתמש בהן.
ג. כתבו את הפונקציה הנדרשת בשאלה הזאת.

תרגיל #6 מיזוג שני תורי-עדיפויות לתוך תור אחד

נתונים שני תורים q1 ו־ q2 של תלמידים, שניהם מסודרים לפי עדיפות כפי שהוסבר בתרגיל #3. כתוב פונקציה בשם MergePriorityQueues שממזגת את שני התורים לתוך התור q1, כך שבסיום התור q1 מכיל את כל התלמידים משני התורים, מסודרים לפי כללי העדיפות. התור q2 ריק. כאשר לתלמיד בתור q2 יש אותו ציון כמו לתלמיד בתור q1 או זוכה לעדיפות גבוהה יותר על פני תלמידים עם אותו ציון בתור q1.

א. כתוב את החתימה של הפונקציה המבוקשת.
ב. כתוב אל האלגוריתם. מירב הנקודות לאלגוריתם יעיל (מהיר).
ג. מהי היעילות של האלגוריתם שכתבת? עליך לנמק בקצרה.
ד. כתוב את הפונקציה.

הערה: בשאלה הזאת לא מתאים להשתמש בפונקציה של שאלה #3. (חשבו מדוע).

תרגיל #7 המספר המינימלי של הזזות בכדי ליצור זהות בין שני תורים

נגדיר פעולת הזזה-מעגלית-בודדת כפעולה שבה מוציאים את האיבר הראשון מהתור ומכניסים אותו לסוף אותו התור.

נתונים שני תורים q1 ו־ q2 של מספרים שלמים. נאמר ששני תורים הם זהים אם – (א) יש להם אותו מספר איברים, (ב) אותם ערכים, (ג) ובאותו סדר.

כתוב פונקציה ששמה הוא MinRotationsToBecomeIdentical. הפונקציה מקבלת שני תורים q1, q2 ומחזירה את המספר K, כך ש- K הוא המספר המינימלי של הזזות מעגליות בודדות שיש לבצע על התור q1 בכדי שהוא יהיה זהה לתור q2. אם לא קיימת אפשרות כזאת אז הפונקציה תחזיר -1.

בסיום הפעולה שני התורים חייבים להישאר ללא שינוי.

א. כתבו אלגוריתם.
ב. מה היעילות של האלגוריתם במקרה הטוב ביותר ובמקרה הגרוע ביותר.
ג. כתבו את הקוד של הפונקציה. זהו את תת־המשימות הדרושות לפתרון, וממשו את הפתרון באמצעות פונקציות עזר.

הנחיה: כשאתם מתארים את יעילות הפונקציה חשבו על המספר N והניחו ש N הוא מספר האיברים בתור הגדול יותר או בשניהם אם הם באותו הגודל. בפתרון שלכם עליכם להסביר (לציין) את ההנחה הזאת.

תרגיל #8 רקורסיה ותורים – סכום המספרים הזוגיים בתור

כתוב פונקציה רקורסיבית שמחזירה את סכום המספרים הזוגיים בתור. הקפד לא לשנות את התור שהפונקציה מקבלת.

הנחיה:

הדרישה לא לשנות את התור, כשמדובר ברקורסיה - זה מצריך תשומת לב, כי תנאי העצירה הוא שהתור ריק. חישבו: איך פתרנו את הבעיה במערכים. מה ישים ומה לא ישים במקרה הזה.