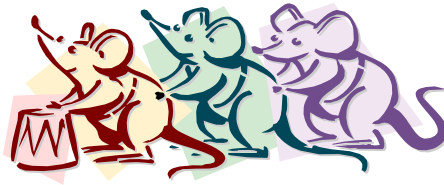
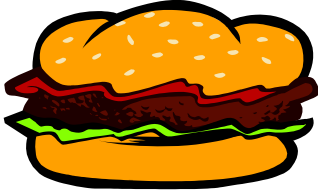


6. מחלקת תור



התבניות שבמחלקה

6.1 - הכנס נתונים לתור

6.2 - הוצא נתונים מתור



תור הוא מבנה נתונים אשר מתאים במיוחד למקרים בהם יש צורך לשמור נתונים ולהוציא אותם לפי סדר שמירתם. מבנה נתונים זה מאופיין בניהול נתונים מסוג FIFO (First In First Out). יש לשים לב להבדל בין מבנה הנתונים תור למבנה הנתונים מחסנית, שהוצג בפרק הקודם. בעוד מחסנית משמשת לניהול נתונים בצורת LIFO, תור משמש לניהול בצורת FIFO. הפעולות בתור מתבצעות דרך שני פתחים נפרדים - "פתח כניסה" שהוא סוף התור, ו"פתח יציאה" שהוא ראש התור. נתון המוכנס בסוף התור "מקודם" בתור עם כל שליפה של נתון מראש התור. לאחר שנשלפו כל הנתונים שלפניו, מגיע הנתון לראש התור.

הדוגמה הבאה מציגה שימוש בתור. כאשר נתון מבוכ המשורטט על לוח משבצות, ונרצה לדעת האם ניתן לעבור דרך 10 משבצות לכל היותר כדי לנוע מנקודת הכניסה למבוכ אל נקודת היציאה ממנו, נשתמש בתור. נתחיל ממשבצת הכניסה וננסה להתקדם משבצת אחת לכל כיוון אפשרי. נשמור בתור כל משבצת אליה נגיע (נשמור את מציני השורה והעמודה שלה), ונצמיד לכל משבצת כזו את הערך "1", המסמן שהיא רחוקה "מרחק של 1" מנקודת הכניסה. לאחר מכן, נבצע בצורה מחזורית את הפעולה הבאה: נשלוף את המשבצת הבאה שבראש התור, וננסה להתקדם ממנה משבצת אחת לכל כיוון אפשרי. נכניס לסוף התור כל משבצת חדשה אליה נגיע, ונצמיד לה ערך גדול ב-1 מן הערך של המשבצת שהובילה אליה. בצורה זו, נוציא ונכניס שוב ושוב משבצות לתור כלשכל משבצת מוצמד ערך, המבטא את מרחקה מן הכניסה למבוכ. כאשר נגיע למשבצת היציאה נדע את מרחקה מן הכניסה ונציג הודעה בהתאם.

בבעיה לדוגמה, המתוארת, שימש התור כמבנה נתונים אשר בסופו מוכנסים שוב ושוב נתונים (אודות משבצות), ומראשו נשלפים נתונים, לפי סדר הכנסתם. השליפה לפי סדר ההכנסה היא חשובה, כיון שהיא מבטיחה שמשבצות קרובות יותר לנקודת הכניסה למבוכ יעובדו לפני משבצות רחוקות יותר. סדר עיבוד זה הינו הכרחי כדי לחשב את מרחקה של משבצת מנקודת הכניסה, כאשר משמעותו של מרחק היא מספר המשבצות בקטן ביותר שיש לעבור בהגעה מנקודה אחת לאחרת.

בתור המוצג בבעיה לדוגמה מוכנס כל נתון רק פעם אחת לתור. לאחר שהוא מוצא/נשלף ממנו, לא יוכנס שוב. כמו כן, הוצאה של נתון (מראש התור) גורמת להכנסה של נתונים חדשים (לסוף התור). ישנן בעיות שבפתרון יתכן שנתון יוכנס ויוצא מן התור מספר פעמים. ישנן בעיות רבות שבהן מתאים להשתמש בתור שהכניסה אליו איננה עקב הוצאה של נתון מראשו, אלא בעקבות הגעה של נתונים לפי סדר מסוים, כגון זמן כרונוולוגי. בפרט, מבנה הנתונים תור יכול להתאים לביצוע סימולציה של תור המתנה לשרות כלשהו, כשסדר ההגעה הכרונוולוגי מכתוב את סדר ההמתנה, והכניסה לתור, ושליפת ממתין מראש התור נעשית בכל פעם שנותן השירות מתפנה. בשאלות בהמשך הפרק ישנן שאלות הפונות לסימולציות כאלה. שאלות אלו כוללות בין-השאר גם היבטים כגון "איחוד תורים" ו"פיצול תורים". התבניות הבסיסיות בהן נעשה שימוש הן התבניות של דחיפת סדרת נתונים לתור וריקון התור מן הנתונים שבו.

יחידות ספריה לטיפול הנתונים **תור** נמצאות באתר תבניות: www.tau.ac.il/~csedu

6.1 - הכנס נתונים לתור

נקודת מוצא: סדרת נתונים, תור מאותחל.

מטרה: הכנסת הנתונים לתור לפי הסדר.

אלגוריתם: הכנס-נתונים-לתור (Q)

$next_element \leftarrow$ איבר ראשון / הסוסה

כאן צריך להוסיף את הסוסה החדשה

הכנס-לתור (Q, next_element)

$next_element \leftarrow$ איבר הבא בסוסה

הערות

- תבנית זו היא תבנית בסיסית של הכנסת סדרת ערכים לתור. סדר שמירת הערכים בתור יהיה לפי סדר כניסתם, כאשר הנתון הראשון יישמר בתור לפני כל הנתונים האחרים, ויהיה הקרוב ביותר לראש התור מבין הנתונים המוכנסים.
- הרחבה של תבנית זו תהיה הכנסת הנתונים בסדרה הנתונה (בנקודת המוצא) ליותר מתור אחד, לפי חוקיות כלשהי. למשל, כאשר יהיו שני תורים יוכנס הנתון הראשון לתור הראשון, הנתון השני לתור השני, הנתון השלישי שוב לתור הראשון, הנתון הרביעי לתור השני, וכך הלאה. בצורה כזו תהיה דחיפה מסודרת לשני תורים. באופן דומה, ניתן להרחיב את השיטה הזו לדחיפה למספר כלשהו K של תורים. שיטה זו רלוונטית כאשר מבצעים למשל סימולציה לניהול מערכת שבה ישנם מספר תורים מקבילים לקבלת שרות.
- וריאציה אחרת לתבנית זו יכולה להיות העברה מתור אחד למספר תורים, או העברה ממספר תורים לתור אחד. במקרה הראשון, תהיה סדרת הנתונים בנקודת המוצא סדרת נתונים השמורה בתור אחד. במקרה השני תהיה סדרת הנתונים בנקודת המוצא הרכבה של נתונים המגיעים בצורה סדרתית כלשהי מקבוצה של תורים.

Pascal

הפעולה מכניסה ערכים מתוך סדרת נתונים לתוך התור Q לפי סדר הגעתם
*) הנחה : סדרת הנתונים אינה ריקה.

(*

```
procedure queue_fill (var Q : queue_type);  
var  
    x : queue_info_type;  
begin  
    queue_init (Q);  
    get_element (x);  
    while x <> end_of_input do  
        begin  
            queue_insert (Q, x);  
            get_element (x);  
        end;  
End;
```

C

הפעולה מכניסה ערכים מתוך סדרת נתונים לתוך התור Q לפי סדר הגעתם
*) הנחה : סדרת הנתונים אינה ריקה.

```
void queue_fill (queue_type *Q)  
{  
    queue_info_type x;  
  
    Q = queue_init ();  
    x = get_element ();  
    while (x != end_of_input)  
    {  
        queue_insert (Q, x);  
        x = get_element ();  
    }  
}
```

6.2 - הוצא נתונים מתור

נקודת מוצא: תור מאותחל.

מטרה: הוצאת הנתונים מהתור.

אלגוריתם: הוצא-נתונים-מתור (Q)

כא Q זרע Q זרע

element ← (Q) זרע-מתור

הערות

- התבנית המוצגת היא תבנית בסיסית של הוצאת כל הנתונים שבתור. הנתונים מוצאים לפי הסדר בו הם הוכנסו. וריאציה של תבנית זו היא הוצאה של חלק מן הנתונים בלבד. למשל, כל הנתונים הראשונים, עד לנתון מסוים, או כל הנתונים שאחרי נתון מסוים, או כל הנתונים שמקיימים תנאי מסוים.

- עבור המקרה האחרון המתואר בהערה הקודמת נשתמש בהוצאה והחזרה לתור. להבדיל ממקרה דומה עם מחסנית, אשר עבורו היה צורך במחסנית עזר, כאן אין צורך בתור עזר. הרעיון הוא שנתוני התור המקורי יישלפו אחד-אחד ויבדקו. נתונים אשר עבורם יתקיים התנאי הנדרש יעובדו ולא יוחזרו לתור. נתונים אשר עבורם לא יתקיים התנאי הנדרש יוחזרו לתור המקורי ויהיו מסודרים בו לפי הסדר המקורי בו היו שמורים מלכתחילה. כלומר, כל נתון שיתברר שהתנאי עבורו איננו מתקיים יוחזר לסוף התור. כיון שהמעבר יהיה על כל נתוני התור, מובטח שבסוף תהליך זה ייוותרו הנתונים שבתור מסודרים לפי סדרם המקורי. הפסאודו-קוד הבא מציג את אופן העיבוד (שים לב לשימוש במשתנה עזר ל"סימון" סוף התור המקורי).

הכנס-לתור (Q, temp)

כא $temp \neq (Q)$ זרע

element ← (Q) זרע-מתור

אם element אז מקיים תנאי אזי

הכנס-לתור (Q, element)

temp ← (Q) זרע-מתור

- המקרה בו נהיה מעוניינים להוציא באופן מיוחד תת-סדרה של נתונים מסוף התור ינוהל באופן דומה לצורה המוצגת בהערה האחרונה, שוב ללא צורך בתור עזר. הנתונים מראש התור יועברו לסופו אחד-אחד, עד אשר "תגיע" תת-הסדרה שיש להוציא לתחילת התור. תת-הסדרה תוצא, ולאחר הוצאתה ישובו הנתונים שלא היה צריך להוציא למקומותיהם המקוריים.

- בשתי ההערות האחרונות, רעיון הפתרון האלגוריתמי משלב שיטה של "סיבוב הנתונים בתור" – נתון בתור "מסתובב" בו על-ידי הוצאתו מן הראש, הכנסתו מחדש בסוף, וקידומו החוזר למקומו המתאים.

<i>Pascal</i>	<pre> (* הפעולה מוציאה את כל הערכים מתוך התור Q *) הנחה: Q תור מאותחל. procedure queue_delete (var Q : queue_type); var x : queue_info_type; begin while not queue_empty (Q) do queue_remove (Q, x); End;</pre>
---------------	---

<i>C</i>	<pre> /* הפעולה מוציאה את כל הערכים מתוך התור Q */ הנחה: Q תור מאותחל. void queue_delete (queue_type *Q) { while (! queue_empty (Q)) queue_remove (Q); }</pre>
----------	--

שאלות

שאלה 6.2.1

בסניף קופת חולים השכונתי נוצר תור ארוך של חולים. הנהלת הסניף שלחה אחות שתחלק את החולים הממתינים לשני תורים: תור של חולים החייבים לראות את הרופאה, ותור של חולים הזקוקים רק למדידת לחץ דם ורישומו לצורך מעקב. פתח אלגוריתם שיקבל כפרמטר את תור הממתינים בסניף Q ויחזיר שני תורים חדשים, כך שבכל תור יהיה סדר הכניסה לרופא לפי סדר הגעת החולים למרפאה.

■

שאלה 6.2.2

כתוב אלגוריתם שיקבל כפרמטר תור Q של מספרים שלמים, ויחזיר את התור לאחר שהוצאו ממנו כל האיברים שערכם שווה ל- k. אין להשתמש במבנה נתונים נוסף.

■

שאלה 6.2.3

כתוב אלגוריתם שיקבל כפרמטר תור של מספרים שלמים Q1. על האלגוריתם לפצל את Q1 לשני תורים כך שכל המספרים הגדולים או שווים ל- k יהיו ב- Q1 וכל האחרים ב- Q2. יש להחזיר את שני התורים כך שהסדר היחסי של המספרים יישמר.

לדוגמה: עבור $k = 10$ ו התור: 4, 7, 12, 65, 23, 5, 8, יישארו ב- Q1 המספרים: 12, 65, 23.

וב- Q2 יהיו המספרים: 4, 7, 5, 8.

■

שאלה 6.2.4

עקב קיצוצים החליט מנהל מפעל לבצע שינויים בכח-האדם. הוחלט לפטר את כל העובדים מתחת לגיל 25 ולהוציא לפנסיה את כל העובדים מעל גיל 65. כתוב אלגוריתם המקבל כפרמטר תור עם נתוני העובדים במפעל (לכל עובד שם, ת.ז, גיל) ומחזיר את התור לאחר השינוי.

■

שאלה 6.2.5

בשאלון בית ספרי התבקש כל תלמיד לציין את המקצוע החביב עליו במיוחד (לכל תלמיד מקצוע אחד בלבד). המנהלת מעוניינת לתת פרס לתלמידים שבחרו במקצוע מדעי המחשב. כתוב אלגוריתם שיקבל זוגות של נתונים: שם התלמיד והמקצוע שבחר, ויחזיר תור Q המכיל את כל אותם תלמידים שבחירתם היתה מדעי המחשב. (יש לשמור על סדר קליטת הנתונים).

■

שאלה 6.2.6

כתוב אלגוריתם שיקבל כפרמטר תור Q ויהפוך את סדר איבריו.

■

שאלה 6.2.7

כתוב אלגוריתם שיקבל כקלט תור Q ממויין בסדר עולה ומספר k, ויחזיר את התור לאחר שהוציא ממנו

את כל המספרים הגדולים מ- k.

■

שאלות סיכום

שאלה 1

במכון הבינלאומי להצפנה התקבלה דרישה להצפנה מיוחדת: הטקסט המוצפן חייב להכיל מספר כוכביות זהה בתחילתו ובסופו. ההודעה המוצפנת אינה חייבת להופיע בתחילת הקלט. ידוע כי טקסט מוצפן בשיטה זו אינו מכיל כוכביות פרט לאלו שבתחילת ההודעה המוצפנת ובסופה, אולם כוכביות יכולות להופיע במקומות אחרים בקלט.

דוגמה: פענוח הטקסט הבא: `How are you****How are you****to*day?` יהיה: `How are you`.
א. פתח אלגוריתם שיקלוט הודעת טקסט (תווי ההודעה מתקבלים בזה אחר זה) וידפיס את ההודעה המפוענחת.

ב. מה יהיה השינוי באלגוריתם שפיתחת אם ידוע כי הקלט יכול להכיל יותר מהודעה מוצפנת אחת?
לדוגמה הטקסט הבא: `Have****abc** a nice**12gd*** day***to*day?` יהיה: `Have****abc** a nice**12gd*** day***to*day?`

יפוענח באופן הבא: `Have a nice day`

- ג. באלו תבניות השתמשת לכתובת האלגוריתמים הנ"ל? סמן אותן בפתרוןך.
 - ד. מהי סיבוכיות כל אחד מהאלגוריתמים שכתבת, כפונקציה של אורך הקלט?
- רמז: השתמש בתור לספירת מספר כוכביות שווה, ולהכנסת תווי הקלט.



שאלה 2

במשחק קלפים "מלחמה" בין שני שחקנים יש שימוש בחפיסת קלפים אחת (בת 52 קלפים). בתחילת המשחק מחזיק כל משתתף בחצי חפיסה אותה קיבל לאחר ערבוב וחלוקה שלבי המשחק:

בכל פעם, חושפים שני השחקנים את הקלף שבראש החפיסה. מי שהקלף שלו גבוה יותר, מעביר את שני הקלפים לתחתית החפיסה שלו.

אם הקלפים בעלי אותו ערך, מכריזים על "מלחמה" ואז כל משתתף מוציא שלושה קלפים מראש חפיסתו ומניח על הקלף הגלוי לפי סדר הוצאת הקלפים. זה שהקלף שבראש הערימה שלו בעל ערך גבוה יותר, מעביר את כל הקלפים משתי הערימות לתחתית חפיסתו. אם שוב יש תיקו, חוזרים על הפעולה.

אם אחד השחקנים נותר ללא קלפים, מוכרז השני כמנצח.

פתח אלגוריתם שידמה את המשחק "מלחמה". האלגוריתם ייצר חפיסה מעורבת של 52 קלפים וישחק משחק שלם שיסתיים בניצחון של אחד המשתתפים. בכל תור ידווח האלגוריתם מהם ערכי הקלפים שחשפו המשתתפים. בסוף המשחק ידווח האלגוריתם מיהו המנצח.

שים לב למקרים בעייתיים כמו מצב של מלחמה שבו לאחד השחקנים אין מספיק קלפים להמשיך. הצעה לדרך ל"ערבוב" הקלפים: ניתן לבנות מערך בן 52 מקומות בו יוכנסו המספרים 1 - 13 4 פעמים. לאחר מכן יוגרלו שני מספרים בתחום 1..52 ותבצע החלפת תוכן התאים במקומות אלו (swap). חזרה על פעולת ההגרלה וה- swap לפחות 100 פעמים, תיצור חפיסה מעורבת.



שאלה 3

לצומת בו רמזור לשני כיוונים – כיוון-1 וכיוון-2 – מגיעות מכוניות בזמנים שונים. הרמזור מתחלף, מאור ירוק בכיוון אחד לאור ירוק בכיוון הנגדי, אחת ל-30 שניות. לאחר 27 שניות של אור ירוק בכיוון מסוים מתחלף האור לכתום, למשך 3 שניות, ואז לאדום ל-30 שניות (בהן האור בכיוון הנגדי הוא ירוק וכתום). מכונית נכנסת למעבר של הצומת רק כאשר האור בכיוונה ירוק והצומת פנוי ממכוניות (קודמתה סיימה לעבור). המעבר של מכונית בצומת אורך 3 שניות. (כלומר, בסה"כ יכולות לעבור עד 10 מכוניות ברצף של 30 שניות מכיוון אחד.) בתחילת השעה 0 נדלק האור הירוק בכיוון-1.

נתונה רשימת קלט של N מכוניות הנכנסות לצומת ($100 < N < 1000$), כאשר עבור כל מכונית חמישה נתונים: מספר המכונית, שעה-דקה-שנייה בה הגיעה והכיוון ממנו הגיעה. הרשימה מסודרת לפי זמנים, בסדר עולה, מן המוקדם למאוחר. הנתון הראשון גדול מן השעה-דקה-שנייה 0-0-0, והנתון האחרון קטן מן השעה-דקה-שנייה 19-30-0. יש להציג כפלט את זמן תחילת המעבר של כל מכונית את הצומת. הפלט יוצג לפי סדר עולה של זמנים.

- א. תאר מאפיינים משמעותיים של הבעיה שעל-פיהם תבחר מבנה נתונים מתאים.
- ב. מהו מבנה הנתונים בו תשתמש לפתרון הבעיה?
- ג. מהן התבניות בהן תשתמש? וכיצד תשלב אותן ביחד?
- ד. נסח פסאודו-קוד לפתרון הבעיה, על-פי מבנה הנתונים והתבניות שבחרת.
- ה. מהי סיבוכיות הפתרון?
- ו. יתכן שנראית לך דרך נוספת לפתרון, תוך שימוש במבנה נתונים שונה (אולי פתרון זה יעיל פחות, אך עדיין רלוונטי). אם כן, ענה עבור דרך נוספת זו על הסעיפים ב-ה.

■

שאלה 4

נתונה מפה של מבוך המשורטטת על נייר משבצות ריבועי. כל משבצת שהיא קיר מסומנת ב-1, וכל משבצת שהיא חלל מסומנת ב-0. כל המשבצות בהיקף המבוך הן קירות, מלבד שתי משבצות, במקומות שונים, שאחת מהן היא הכניסה למבוך והשניה היא היציאה ממנו. ישנם במבוך מספר מסלולים שונים למעבר מנקודת הכניסה לנקודת היציאה (דרך משבצות מסוג חלל).

יש להציג כפלט את מספר המשבצות במסלול הקצר ביותר מן הכניסה ליציאה.

- א. תאר מאפיינים משמעותיים של הבעיה שעל-פיהם תבחר מבנה נתונים מתאים.
- ב. מהו מבנה הנתונים בו תשתמש לפתרון הבעיה?
- ג. מהן התבניות בהן תשתמש? וכיצד תשלב אותן ביחד?
- ד. נסח פסאודו-קוד לפתרון הבעיה, על-פי מבנה הנתונים והתבניות שבחרת.
- ה. מהי סיבוכיות הפתרון?
- ו. יתכן שנראית לך דרך נוספת לפתרון, תוך שימוש במבנה נתונים שונה (אולי פתרון זה יעיל פחות, אך עדיין רלוונטי). אם כן, ענה עבור דרך נוספת זו על הסעיפים ב-ה.

■

שאלה 5

באליפות העולם בריצת 100 מטר מחולקים הרצים הרבים בסיבוב הראשון לקבוצות של 8 רצים לכל מקצה. מכל מקצה עולים לשלב הבא שני הטובים ביותר. זוגות המנצחים ב-4 מקצים סמוכים מקובצים למקצה אחד בסיבוב הבא. כך מתקדמת התחרות סיבוב אחר סיבוב, עד שלב הגמר, שהמנצח בו הוא המנצח בתחרות כולה. מספר הרצים התחילי גדול מאד, והוא חזקה של 8 (כך מובטח שבכל מקצה, בכל סיבוב יהיו 8 רצים). מספר הסיבובים עד שלב הגמר גדול.

נתונה רשימת קלט, המתארת את תוצאות התחרות. תחילה נתונים כל המקצים של הסיבוב הראשון ברצף (שאורכו כמספר המקצים בסיבוב כפול 8), אחריו כל המקצים של הסיבוב השני, וכך הלאה, ולבסוף המנצח. כל רץ מצוין על-ידי מספר סידורי, שהוא מספר שלם חיובי כלשהו. כל 8 רצים במקצה של סיבוב הם ארבעה זוגות המנצחים בארבעה מקצים שהופיעו ברצף בסיבוב הקודם (למשל, המקצה הראשון בסיבוב השני מורכב מארבעת זוגות המנצחים בארבעת המקצים הראשונים בסיבוב הראשון). בקלט, מופיע הנתון 0, בסוף כל סיבוב ("מפריד" בין סיבובים).

יש להציג כפלט את כל המקצים, שורה אחר שורה, לפי הסיבובים השונים, כך שבשורה של כל מקצה יוצגו 8 המתחרים בו, ואחריהם שני המנצחים בו. עבור מקצה הגמר יוצג רק המנצח בו.

א. תאר מאפיינים משמעותיים של הבעיה שעל-פיהם תבחר מבנה נתונים מתאים.

ב. מהו מבנה הנתונים בו תשתמש לפתרון הבעיה?

ג. מהן התבניות בהן תשתמש? וכיצד תשלב אותן ביחד?

ד. נסח פסאודו-קוד לפתרון הבעיה, על-פי מבנה הנתונים והתבניות שבחרת.

ה. מהי סיבוכיות הפתרון?

ו. יתכן שנראית לך דרך נוספת לפתרון, תוך שימוש במבנה נתונים שונה (אולי פתרון זה יעיל פחות, אך עדיין רלוונטי). אם כן, ענה עבור דרך נוספת זו על הסעיפים ב-ה.

