רשימה

מבוא כללי

תפיסת המושג רשימה

כאשר מתייחסים לרשימה באופן כללי, ניתן לדבר על תפיסת שונות. התפיסה האינטואיטיבית של הרשימה היא גישת המיקום המספרי של האיברים. מקום ברשימה נתפס כאינדקס = מקומו מספרי של האיבר. תלמידים נוטים באופן טבעי לגישה זו. חשוב להבין שזו אינה טעות וכי גישה זו קיימת גם בספרות האקדמית. גם המחלקה LinkedList הנמצאת בJavaAPI, מתייחסת למקום ברשימה כאינדקס. על מנת לשפר את היעילות היא משתמשת ברעיון מופשט מתקדם: איטרטור (Iterator).

הגישה שאנו מציגים בפרק היא **גישה לפי מקום בזיכרון**: **positional** - גישה זו מתייחסת למקום מסוים ברשימה, לא כאינדקס, אלא כהפניה למקום בזיכרון. קיימות כמה דרכים לנהל מקום כזה: אחת הדרכים היא להצביע למקום מסוים עם הפנייה (כפי שאנחנו עושים בפרק). דרך אחרת, היא לבנות אובייקט שאחראי לניהול המקום בזמן הריצה (כך פועל האיטרטור אך אנו לא נוכל להשתמש ברעיון זה לפני שהתלמיד יתקדם רבות בלימודיו).

הייצוג שאנו מציגים לרשימה בפרק מכונה אצלנו: שרשרת חוליות והוא מכיל למעשה גם מעטפה לשרשרת החוליות. ניתן בהחלט להציג ייצוג של רשימה רק כשרשרת חוליות ללא המחלקה העוטפת. לייצוג זה יתרונות שונים (אם כי גם חסרונות) וניתן בכל זאת לתרגל אותו, להדגים עליו סריקה רקורסיבית של רשימה ועוד.

בפרק אנו מציגים את המקום של איבר ברשימה בעזרת המחלקה Node.
החיסרון של גישה זו הוא בכך שהרשימה אינה טיפוס נתונים מופשט (טנ"מ). אופן הייצוג של מקום ברשימה חשוף למשתמש, בזמן שינוי הייצוג משתנה ממשק הרשימה והיא חשופה לשינויים במבנה שלה באופן בלתי רצוי: דרך פעולות החוליה ניתן בקלות לפגוע בהגדרת הרשימה וליצור רשימה מעגלית. הרשימה אינה יכולה להגן על עצמה בגלל שהחוליה שלה חשופה.
סעיף **ט** בפרק מסכם באופן ברור את הרעיון שרשימה היא **מבנה נתונים** אך אינה יכולה להיחשב כ**טיפוס נתונים מופשט**. אין צורך להתעמק עם תלמידים ממוצעים בהגדרות המדויקות של "טיפוס נתונים מופשט" מול "מבנה נתונים" אך הדברים מובאים במפורש וברמיזות בפרקים, וכיתות מתקדמות ומורים המחבבים את הנושא יכולים בהחלט להתעכב על הגדרות אלו. לטובת התעמקות זו שולבו תרגילים שונים בסוף הפרק העוסקים בהגדרת טיפוס נתונים מופשט ובמבני נתונים. כיתות שדילגו על ההבחנות יכולות לבצע את התרגילים מבלי להתעכב על פרטי ההבחנות בסעיפים הללו.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

הצעה למהלך הוראה

1. דיון על המושג המופשט רשימה כפי שהוא מוכר מחיי היום יום. לדבר על הגמישות של המבנה לעומת מערך.
2. אפיון המושג חוליה ושרשרת חוליות. האם נדרש משהו שינהל את שרשרת החוליות? רמיזה על המבנה הרקורסיבי של שרשרת זו.
3. שימוש ב-UML ודרך כתיבה סטנדרטית של ממשקים. יש חשיבות מרובה להרגיל את התלמידים לרכז את כל המידע שיש בידיהם בעזרת איור בשפת UML ולהיפך: לפענח את כל האינפורמציה הטמונה באיור כזה. שפה אחידה זו מאפשרת העברת ידע רב בצורה חד משמעית ומתומצתת. כך יש גם להתעכב בתרגילים השונים על השימוש הדו כיווני באיורים אלו.
4. הגדרת המחלקה חוליה. דיון על תכונות המחלקה ופעולותיה. אם מתעורר צורך ניתן לדון מדוע לא מוגדרת פעולה בשם getPrev() (סימטרית ל-getNext()) בממשק החוליה.
5. אפיון המושג רשימה כמעטפת של שרשרת החוליות. המעטפת אחראית על הוצאת חוליות, הכנסת חוליות חדשות וסריקת המבנה. בהמשך לסעיף 2 כדאי לשוב ולציין שלא תמיד נבחר לעטוף את שרשרת החוליות במחלקה עוטפת "רשימה".
6. משמעות המושג "מקום ברשימה". מדוע הפניה לחוליה יכולה למלא תפקיד זה. ניתן להתקדם לדיונים בהגדרות אחרות למושג 'מקום'.
7. דיון בפעולות הכנסה והוצאה. כאן ניתן להדגיש כי קיימות מספר חלופות להגדרת הפעולות. חלק מהדיונים בהגדרת הפעולות נסקרים בתוך הסעיפים השונים. ניתן לדון בערכי החזרה שונים: למשל האם להחזיר את מיקום האיבר הבא בפעולה insert, ניתן לדון בפעולות הנחוצות: האם להגדיר פעולה getPrev()? האם להגדיר פעולה מיוחדת בשם insertFirst(…)? מהו המיקום אותו כדאי להעביר כפרמטר לפעולה insert, ומהו המיקום שאותו כדאי להעביר כפרמטר לפעולה delete. כל השאלות האלו יכולות להיות פתוחות לדיון, והדבר תלוי ברמת הכיתה.

   תרגול הרעיון שניתן לשנות את פעולות הממשק ומה תהיה עלות השינוי ומשמעותו – יכול להיעשות בעזרת תרגילים המופיעים בסעיף: פעולות פנימיות.

   לפניכם סקירה קצרה של פעולות הממשק כפי שהן מופיעות בממשק המחלקה:

   i. פעולת ההוצאה

   יעילות פעולת ההוצאה היא O(n), כיוון שעל מנת להוציא איבר ממקום מסוים, יש למצוא את האיבר הקודם לו. חיפוש האיבר הקודם, מתבצע על ידי סריקת הרשימה בעזרת הפנייה, החל מראשה. ניתן כבר בשלב הזה לעודד את התלמידים לחשוב, איך ניתן ליעל את הפעולה על ידי שינוי הייצוג (לקראת התרגיל של רשימה דו כיוונית).

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

ii. פעולת הכנסה

הפרמטר הנשלח:

לפי מה שמפורט בפרק, ניתן להגדיר את פרמטר המקום שאותו מקבלת הרשימה באופן שונה. ניתן להגדירו כמקום שהערך ייכנס אחריו, או מקום שהערך ייכנס לפניו. כדאי לדבר עם התלמידים על ההשלכות הנובעות מכל בחירה על השימוש ברשימה. למשל, החלטה על הכנסה לפני המקום הנתון, מחייבת לחפש את המקום הקודם לנתון עבור כל הכנסה, כלומר: סיבוכיות פעולת ההכנסה הייתה הופכת ל- O (n).

חשוב לדבר על מקרי קצה, למשל: הכנסה למקום הראשון. ניתן להגדיר פעולת הכנסה פרטית למקרה זה אך אנו החלטנו על הצורה המיוחדת (הפחות אינטואיטיבית), של זימון הפעולה במקרה זה עם ערך **null**. (תרגול האפשרות של פעולת הכנסה אחרת בסעיף: פעולות פנימיות.

ערך ההחזרה:

ניתן היה לא להחזיר ערך כלל, אך בחרנו להחזיר את המקום של החוליה שהוכנסה. ניתן להגיד לתלמידים כי הדבר נעשה מתוך שיקולי נוחות לצורך הסריקה. חשוב ללמד את התלמידים להשתמש בערך ההחזרה (בעזרת הדוגמאות המופיעות בפרק).

דוגמה מעניינת:

הכנסת 4 איברים ראשונים לרשימה של מחרוזות:

```
List<String> lst = new List<String>();
Node<String>  p = null;
for (int i =0; i<4; i++)
{
        String str = IO.readString()
        p = lst.insert (p, str);
}
```

כדאי לדון על אופן קידום ההפניה. כאשר פעולת ההכנסה מוגדרת כך ההכנסה למקום הראשון נראית בדיוק כמו כל הכנסה ואכן יש רק פעולה אחת להכנסה בכל מקום. הכנסה ללא קידום ערכו של p אפשרית ותקינה אך משמעותה שונה: אנו נכניס כל איבר נוסף לראש הרשימה!

iii. הפעולה toString() מניחה למעשה שהפעולה מוגדרת לכל עצם (דבר שהוא כמובן נכון!!) ולכן אפשר להגדיר אותה כפעולה פנימית בתוך מחלקה גנרית, למרות שהיא מתייחסת לערך של האיבר! לא כדאי לדבר ולדון על נושא זה אלא אם הדבר עלה בכיתה על ידי התלמידים או בכיתות טובות במיוחד.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

7. תרגול השימוש ברשימה, יצירת הרשימה, הכנסת האיברים אליה והוצאתם, כל אלו מאפשרים להפנים את רעיון מבנה הנתונים ופעולותיו לפני שפונים למימושו ולכן לא כדאי לדלג עליהם.
8. מימוש מלא של שתי המחלקות Node ו־List.
9. פעולות פנימיות וחיצוניות – יש להדגיש שהפעולות הפנימיות לא יוכלו לבצע חישובים והשוואות על הערכים השמורים ברשימה הגנרית כיון שלא לגבי כל העצמים יש משמעות והגדרה שווה לפעולת השיוויון או ה־compareTo, לפעולת הסכימה + וכד׳. לכן פעולות המצריכות התייחסות לערכי הרשימה יוגדרו מראש כפעולות חיצוניות על רשימות קונקרטיות. יוצאת דופן היא הפעולה toString() – ראו בפירוט לעיל.

   כל הפעולות החיצוניות על רשימה ללא יוצא מהכלל יוגדרו על רשימות קונקרטיות ולא יוגדרו כגנריות, גם אם הן אינן מתייחסות לערכי הרשימה. לדוגמה: הפעולה אורך רשימה. הסיבה להחלטה זו: א. שמירה על קו פעילות אחיד תמנע הסתבכות של התלמידים. ב. לפעולות חיצוניות גנריות יש סינטקס מיוחד:

   public static <T> **int** getLength (List<T> lst)

   על מנת שלא להיכנס לתסבוכות מיותרות ניקח מהגנריות רק את המינימום העוזר לנו ונוותר על כל השאר לשלבים מתקדמים יותר של הוראה.
10. דיון במבנה של ממשק הרשימה. בשונה מטיפוס המחסנית, אין לרשימה ממשק סגור ואחיד. סעיף **ד.3.** דן בנושא. חלק מהדברים נאמרו כבר בפרק המחסנית אך שייכים ומורחבים בפרק זה.
11. הפעולות המתקדמות מאפשרות לחדד כיווני פעולה שונים עם רשימות: כמה הפניות לרשימה, עבודה עם מספר רשימות במקביל ועוד.
12. ייצוג טיפוסי נתונים מופשטים בעזרת מבני נתונים. גם אם המושג טנ״מ לא נלמד ביסודיות בכיתה כדאי בכל זאת לאפיין את הרשימה כמבנה נתונים, המאפשר שמירה של אוסף נתונים וטיפול בו, אך אינו מוגדר באופן סגור וברור. כך הוא יכול לשמש לצורך ייצוג נתונים של אוספים אחרים כאשר חלק מהפעולות של הרשימה שאינן מתאימות לטיפוס הנתונים, ׳מוחבאות׳ והמשתמש לא יכול להשתמש בהן. גם נושא זה ניתן לתרגול בעזרת השאלות המופיעות בסעיף: טיפוסי נתונים מופשטים.
13. חוליות בעלות שתי הפניות הן הרחבה של החידוש שברעיון המבנה המשורשר. הוא נסקר בקצרה בפרק ומתורגל בהמשך. יש חשיבות מרובה להתעכב על הנושא שכן הוא מאפשר גמישות וניצול רב יותר של מבנה הנתונים ׳חוליה׳ שנלמד בפרק.
14. סקירה של נושא טיפוס נתונים מופשט בהקשר של רשימה. הדיון המעמיק אינו מומלץ לכל הכיתות ויעשה על פי שיקול דעתכם.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

הערות חשובות:

- הרשימה אינה טיפוס נתונים מופשט:

1. גם אם לא נכנסים לדיוקים העדינים של הגדרת טיפוס נתונים מופשט כדאי לחדד אצל התלמידים את התחושה של הסתרת המימוש, שמירת הממשק גם בעת שינוי הייצוג והיחודיות של הטנ"מ כשומר באופן כזה על בטיחותו והגדרתו.
2. הייצוג שלה חשוף למשתמש. החוליה, שהיא תכונה פנימית של הרשימה אינה מוסתרת וניתן לשנות אותה גם ללא פעולות הרשימה.
3. הרשימה אינה יכולה לשמור על המבנה הפנימי התקין שלה. ניתן לבצע פעולת setNext בצורה כזאת, שייווצר מעגל בתוך הרשימה.

ג'אווה מתמודדת בהצלחה עם בעיה זו בעזרת ה-Iterator שאינו מתאים לתלמידינו בשלב זה של לימודיהם אך כדאי שיבינו את הבעיה ויכירו אותה, ובנוסף יבינו שהרשימה אינה טנ"מ בגלל בעיה זו.

- למחלקה Node ישנה חשיבות גדולה מאוד בפני עצמה. ניתן אף לטעון כי היא מבנה הנתונים המרכזי בפרק: מבנה מקושר. לעומת מערך היא מאפשרת תחזוקת זיכרון גמישה ולא מוגבלת בגודל. התעסקות עם שרשרת חוליות ללא מחלקה עוטפת מאפשרת לחשוף את התלמידים למבנה רקורסיבי פשוט, ולתרגל איתם תכנות רקורסיבי כמבוא טוב לעצים.
- למרות שהרשימה היא גנרית והטיפוס שלה הוא טיפוס כללי T, אנחנו יכולים להגדיר את הפעולה ()toString כפעולה פנימית של הרשימה. זאת כיוון שבג'אווה קיימת הנחה בסיסית כי לכל טיפוס מוגדרת פעולת ()toString. (בעצם כל טיפוס יורש אותה מטיפוס האב שלו Object). ואגב – כאשר אנו כותבים UML או ממשק של הפעולה איננו מציינים את אופן הכתיבה המדויק שלה בעזרת override.
- כתיבת אלגוריתמים בתרגילים – תעשה באופן שוטף וכללי ותסרטט את מבנה הפתרון ללא כניסה למשתנים וקידומם באופן מפורט. תרגול הנושא של כתיבה אלגוריתמית שכזו נמצא בחלק מן התרגילים ומודגם לאורך הפרק.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

תשובות לשאלות המחשבה בגוף הפרק (חלקי)

סעיף א.1 עמוד 4 – יש לשים לב שלמרות שיצירת מעגל ברשימה סותר את הגדרתה כאוסף לינארי, ה"רשימה" אינה יכולה להתגונן נגד קלקול המבנה. מכיון שהרשימה אינה טיפוס נתונים מופשט היא אינה יכולה למנוע חבלות במבנה שלה דרך אוסף הפעולות המוגדרות עליה (שונה מכך היא המחסנית שאין שום דרך לקלקל את המבנה שלה בעזרת הפעולות שלה ואכן היא טנ"מ).

עמוד 14 – toString של הרשימה ניתן להיכתב בשתי דרכים. זו הזדמנות להציג את האופן הרקורסיבי של טיפול ברשימה דרך אוסף החוליות ולהדגיש שהן חשופות (זה לא טיפוס נתונים מופשט).

קשיים צפויים:

- ההבנה כי מקום ברשימה הוא הפניה לחוליה ברשימה.
- שימוש בפעולות הכנסה והוצאה ובעיקר נושא הערך המוחזר מפעולות אלו.
- ההבדל שבין שרשרת חוליות לבין רשימה שבתוכה שרשרת חוליות. כדאי בכל זאת למקד את הדיון בהבדלים, לבחון את אופן הציור של מבני הנתונים כדי לחדד את ההבדל ביניהם ולתרגל את שתי צורות הייצוג כדי להכשיר את התלמידים להבין שרעיון מופשט של טיפול באוסף (הרשימה) יכול להיות מיוצג וממומש באופנים שונים על ידי מבני נתונים שונים שלכל אחד מהם יתרונות וחסרונות. ההתאמה והבחירה במבנה הנתונים תעשה בהתאם לדרישות או על פי מהות הבעיה שבה ניתן דגש או עדיפות לנקודות מסוימות שיכוונו אותנו לבחירה ולהעדפה מסוימות.
- הבנת הרעיון שפעולה פנימית אינה יכולה לפנות לערך בתוך הרשימה, לעסוק בו ולבצע עליו חישובים והשוואות בגלל הגנריות של החוליה והרשימה.
- הבנת ההבדל בין רשימה גנרית לבין רשימה קונקרטית.

תרגילים

- התרגילים מחולקים לנושאים ומוצגים בסדר מתפתח על פי התכנים של הפרק ולפי סדר הקושי וההפשטה הנדרשים לפתרונם. אין כמובן צורך לפתור את כל התרגילים אך כדאי לוודא שהתלמידים התנסו בכל סוגי התרגול.

א. פעולות חיצוניות (9-1)

מבחינה דידקטית נראה ששימוש בעצמים ומחלקות צריך להקדים את הטיפול בייצוג מחלקות ומימושן. לכן מופיע נושא הפעולות החיצוניות כראשון בפרק ובתרגול. בפעולות החיצוניות התלמיד יכול להשתמש רק בממשק של הרשימה.

שאלה 5 – מאפשרת לשלב בין טיפול ברשימה ושילוב מחלקה גרפית שהתלמידים מכירים, כך שהתוצאה המתקבלת - משמחת.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

שאלה 8 – פעולה בונה מעתיקה – בגלל הגנריות אין לנו ברירה אלא להגדיר פעולות בונות מעתיקות כפעולות חיצוניות הפועלות על עצמים קונקרטיים בלבד. כך תהיה לנו דרך להעתיק את הערכים השמורים בחוליה או ברשימה באופן מדויק. ניתן לפתח דיון כדי לחדד את נושא ההפניות ולהראות לתלמידים מה יקרה אם נרצה להגדיר את הפעולות הבונות כפעולות פנימיות של המחלקות (אנו נצטרך לוודא שלכל עצם יש פעולה מעתיקה או שייוצר מצב של עצם חדש המכיל הפנייה לאותו ערך (עצם) השמור בשדה הערך!)

ב. שאלות מעקב (10-12)

תרגילים אלו בדומה לשאלות על פעולות חיצוניות, מתרגלים את ההבנה של הממשק והשימוש בו.

פתרונות חלקיים:

שאלה 10 – הפעולה משרשרת את שרשרת החוליות השנייה להמשך שרשרת החוליות הראשונה. השרשרת הראשונה משתנה ממש בפועל ולכן אין צורך להחזיר שרשרת חוליות חדשה בתור ערך החזרה (**void**). ניתן לקיים דיון עם התלמידים כיצד ניתן למנוע את 'קלקול' השרשרת הראשונה ולהחזיר שרשרת חוליות חדשה, משורשרת, בעוד שתי המקוריות לא נפגעות. ניתן לדון גם בהבדלים ברמת הסיבוכיות...

שימו לב שלמרות שהשאלה עוסקת בשתי רשימות, הן מיוצגות כשרשרות של חוליות ללא המחלקה העוטפת List, כדי לאפשר התעסקות בפועל עם החוליות של שתי ה"רשימות".

שאלה 11 – התוכנית מחזירה 2 עבור רשימה שבה מספר זוגי של איברים ו-1 עבור רשימה בה מספר החוליות אי זוגי.

שאלה 12 – הפעולה בודקת אם הסימן של הערכים בחוליות משתנה. שימו לב שהתוכנית מתעסקת בשרשרת חוליות ולא במעטפת של הרשימה. התוכנית לוקחת את הסימן הראשון והולכת לפיו לאורך שרשרת החוליות. אם המכפלה הופכת שלילית בשלב כלשהו זהו סימן שהתחלף סימן והתוכנית תחזיר שקר. יש לשים לב שהנחת בסיס היא שהפרמטר קיים ואינו **null**.

ג. פעולות פנימיות (13-19)

הפעולות הפנימיות מתרגלות ייצוג של מחלקה וגישה ישירה לייצוג זה. יש כאן "החלפת כובעים": בפעולות הפנימיות התלמיד הוא מתכנן המחלקה ויש לו נגישות לייצוג, לעומת זאת בפעולות החיצוניות התלמיד יכול להשתמש רק בממשק של הרשימה. אם מתרגלים רק את הפעולות החיצוניות, התלמיד אינו חייב להכיר את התכונות הפרטיות של הרשימה ואת המימושים השונים. בסעיף זה נדרש התלמיד להכיר ביסודיות את דרך ייצוג המחלקה.

שאלות 13-16 – בשאלות אלו אין צורך לשנות את הייצוג המקורי של הרשימה.

שאלה 17 – בשאלה זו נעשה תרגול של שינוי הייצוג של החוליה. שאלה זו חשובה לצורך הרחבת הרעיון של מבנה נתונים המכיל הפניות.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

שאלות 18-19 – בוחנות דרכים אחרות להגדרת פעולות הממשק. נחזור ונדגיש כי פעולות הרשימה אינן מוחלטות וניתן לפגוש גרסאות שונות של ממשקים בספרות.

לגבי שאלה 18 – insert: תרגיל זה הוא תרגיל קל, אך הוא גורם לתלמיד לעבור על מימוש הפעולה הקיימת ולהבין אותה. ניתן לקיים דיון ולהשוות בין שתי החלופות: פעולת הכנסה אחת או שתי פעולות הכנסה נפרדות. מחד המימוש של הפעולה המוגדרת בממשק שבפרק מורכב יותר, כי הוא מכיל את כל המקרים בפעולה עצמה. מאידך השימוש בה נוח יותר, כיוון שפיצול לשתי פעולות לא מאפשר הכנסה אחידה לכל האיברים.

לגבי שאלה 19 – remove: ניתן לדון ביעילות הפעולה, לעומת הפעולה הקיימת. יעילות ההוצאה בהגדרה החדשה היא $O(1)$. אך לעומת זאת ישנם תרגילים שפחות נוח לפתור אותם עם הפעולה בהגדרה החדשה. למשל:

"כתוב פעולה המקבלת רשימה כלשהי ומוציאה ממנה את כל האיברים בעלי ערך זוגי".

ניתן לתת לפתור תרגיל זה לתלמידים בשתי הגרסאות של הפעולות ולהשוות ביניהם.

ד. טיפוסי נתונים מופשטים (20-24)

בתרגילים אלו אנו מתעסקים בטיפוסי הנתונים המופשטים כמתכנתים (המממשים של המחלקות). אנו משנים מימושים כדי לשפר יעילות וכד', אך עדיין הטנ"מ נשאר טנ"מ. מצד המשתמש אין שום דרך לדעת מה המימוש והאם הוא השתנה, כל זמן שאנו אכן שומרים על אותו ממשק.

סוג תרגילים זה ממחיש את הרעיון של טיפוס נתונים מופשט. ניתן לדון עם התלמידים בשאלה האם המימוש משנה את הממשק או שאינו משנה את הממשק. כדאי מאוד לבצע לפחות חלק מתרגילים אלו, כיון שאותם טיפוסים ניתן לייצג גם באמצעות עצים, ואופני הייצוג השונים מדגישים את החשיבות של בחירת הייצוג לנושא היעילות.

המלצה כללית – להוסיף לכל אחת מהשאלות על הטיפוסים שני סעיפים לדיון:

i. האם הטיפוס הוא טיפוס נתונים מופשט?

ii. האם ניתן להכליל את הטיפוס לטיפוס גנרי במקום שיהיה בעל אפיון מסוים?

שאלה 20 – מחסנית. ניתן להראות לתלמידים ייצוג המחסנית בשתי דרכים שונות אינו משנה את הממשק שלה. הדבר נובע מכך שהמחסנית היא טיפוס נתונים מופשט.

כדאי לקיים דיון איזה מהמימושים נוח יותר. וכן להשוות את שני המימושים למימוש באמצעות מערך. גם אם לא נכנסים לדיוקים העדינים של הגדרת טיפוס נתונים מופשט כדאי לחדד אצל התלמידים את התחושה של הסתרת המימוש, שמירת הממשק גם בעת שינוי הייצוג והיחודיות של הטנ"מ כשומר באופן כזה על בטיחותו והגדרתו.

שאלה 21 – תור. הדרישה היא לממש את התור באמצעות חוליה בלבד, על מנת לתרגל מחלקה זו ללא הרשימה. תיקון היעילות יעשה אם התלמידים יגדירו בייצוג של המחלקה הפנייה לסוף הרשימה.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

שאלה 22 – קבוצה. לא ניתן לממש קבוצה גנרית באמצעות הכלים הקיימים ברשותנו כרגע, כיוון שבמחלקה זו דרוש שימוש בפעולת ההשוואה, ואין לנו כלים להגדיר מחלקה רק עבור טיפוס הניתן להשוואה. במידה והזמן קצר, ניתן לוותר על מימוש פעולות האיחוד והחיתוך. במידה ומחליטים לממש פעולות אלו יש להסביר אותן היטב בכיתה ולהיעזר בדוגמאות.

קיימות מספר דרכים לממש את הקבוצה: באמצעות חוליה, באמצעות רשימה או באמצעות מערך. מימוש הקבוצה באמצעות חוליה או באמצעות רשימה אינו משנה את היעילות: ההבדל ביניהם הוא רק בנוחות כתיבת הפעולות.

במימוש של המערך קיימות כמה אפשרויות: כיוון שאין שום סדר בין איברי הקבוצה, ניתן למחוק את האיברים של הקבוצה באופן שישאיר "חורים" במערך, למשל על ידי הסימון שלהם כפנויים באמצעות ערך בוליאני כלשהו. במקרה כזה ההכנסה לקבוצה תתבצע תמיד ב$O(n)$, כיוון שגם ברשימה יש צורך לעבור על כל האיברים על מנת לבדוק האם איבר שמוכנס קיים ברשימה, וגם במערך יש צורך לבצע זאת.

שאלה 23 – אוסף ממוין. שאלה זו מכילה נקודות התייחסות מעניינות רבות ולכן מומלץ ביותר לבצע אותה ביחד בכיתה תוך דיון וליבון כל הנקודות הבאות:

אוסף ממוין לא ניתן לממש כאוסף גנרי, כיוון שעל ערכי האוסף חייבת להיות מוגדרת פעולת ההשוואה. ניתן לדון במימושים שונים של האוסף הממוין: חוליה, רשימה או מערך. בעתיד נדבר על מימוש נוסף מתקדם.

יש לשים לב שאין שום חיוב לשמור את הנתונים של האוסף באופן ממוין בדרך הייצוג שלהם. למשתמש החיצוני אין דרך לדעת כיצד שמורים הנתונים והאם הסדר ביניהם זוכה לייצוג פנימי או בא לידי ביטוי רק בתוצאת הפעולות המתאימות (getAll למשל). זוהי אחת התוצאות של היות האוסף טיפוס נתונים מופשט. בכל זאת על ידי הטלת הגבלות שונות בנושא היעילות אנו יכולים לכוון את המתכנת לדרך ייצוג מסוימת.

לגבי הסיבוכיות: פעולת ההכנסה פחות נוחה במערך מאשר ברשימה או בשרשרת חוליות, כיוון שצריך להזיז את כל איברי המערך על מנת להכניס איבר למקום מסויים. יעילות פעולה זו היא $O(n)$. לעומת זאת במערך הפעולה exists יעילה, כיוון שניתן לבצע אותה ב-$O(\log n)$ על ידי ביצוע חיפוש בינרי (חציית המערך בכל פעם). ברשימה לעומת זאת, ההכנסה של האיבר היא ב-$O(n)$ כיוון שצריך לחפש את המקום הנכון להכנסה, וגם הפעולה exists היא ב-$O(n)$, כיוון שאין לנו ידע לגבי אורך הרשימה. אם מחזיקים כתכונה את גודל הרשימה, אז ניתן לבצע חיפוש בינרי על הרשימה בדומה למערך.

פעולת המחיקה אף היא ב-$O(n)$, כיוון שמציאת האיבר היא ב-$O(n)$.

כאשר נממש את האוסף הממוין באמצעות עץ חיפוש בינרי בעתיד, ההכנסה תהיה ב-$O(\log n)$, וגם חיפוש יהיה ב-$O(\log n)$. המחיקה היא יותר קשה אלגוריתמית, ולכן בייצוג של עץ היא תהיה סעיף רשות.

הנחות ובדיקות: שאלה זו מהווה הזדמנות טובה לדון מחדש בנושא של **הנחות** שצריכות להופיע בתיעוד אך אינן נבדקות בקוד, לעומת בדיקות קצה שאותן יש לבדוק ולממש בקוד עצמו (למשל,

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

בפעולה getNextValue (**int** x), ההנחה היא ש-x קיים באוסף ואין צורך לבדוק זאת אך יש לבדוק אם אכן יש לו עוקב ולפעול בשתי דרכי פעולה שונות בהתאם למצב).

עצם ריק: הפעולה getAll על אוסף ממוין **ריק** תחזיר עצם מטיפוס מערך אך העצם יהיה ריק. אם נשאל לגבי אורכו של המערך המוחזר – נקבל 0. כדאי לחדד עם התלמידים את הרעיון שקיים עצם למרות שהוא ריק ולהדגים להם.

<u>שאלה 24</u> – וקטור. ניתן לפתח דיון על ההבדל בין הרשימה, שבה המקום הוא הפניה לזיכרון, לבין וקטור שבו המקום הוא אינדקס מספרי.

ה. מבני נתונים (25)

ייצוג רשימה בעזרת מערך – שימוש במבנה נתונים מוכר לצורך ייצוג מבנה הנתונים החדש. תכונה של המחלקה רשימה, היא למעשה המאפיין שלה המגדיר את אופן שמירת הנתונים. בייצוג הרגיל של רשימה, המאפיין של המחלקה היה הפנייה אל החוליה הראשונה אך כעת האפיון מראה על דרך שמירת נתונים אחרת!

ברור שהרשימה אינה טיפוס נתונים מופשט וההוכחה היוצאת מתרגיל זה היא הצורך לשנות ממש את פעולות הממשק עם שינוי הייצוג! גם כיתות שלא העמיקו בנושא טנ"מ כדאי ברמה הזו של אי הפרדה בין מימוש לממשק לדון ולדייק.

ו. מידול (26)

שאלות מידול הן שאלות קשות בעיקר מכיון שאין דרך אחת מוחלטת למדל בעיות וכדי לעבור ולבחון את הצעות התלמידים השונות נדרשים זמן ופתיחות. בכל זאת מומלץ להתנסות לפחות בשאלה אחת בסגנון זה שכן כאן נחשפים ומודגשים הנושאים החשובים המאפיינים את התכנות מונחה העצמים: אלו ישויות (עצמים) משתתפים בפתרון, הגדרת העצמים, הקשרים בין המחלקות, ההבחנה בין ממשק לייצוג ומימוש ועוד.

ניתן לייצג ספר טלפונים באמצעות מערך. החיסרון של מימוש שכזה יהיה שההכנסה וההוצאה של כל כניסה בספר יהיו פעולות לא פשוטות. כמו כן גודל הספר יהיה מוגבל.

ייצוגים אפשריים נוספים: רשימה או שרשרת חוליות, וכן אוסף ממוין מתרגיל 23.

הרחבה משמעותית ומוסברת לשאלת ספר הטלפונים נמצאת בפרק 8=מפה וכדאי לא להתייחס לתרגיל זה כאן אלא בפרק מפה על פי המדריך למורה במקום.

ז. שאלות כלליות (27)

זוהי דוגמה לשאלת בגרות שהותאמה לרוח היחידה המחודשת. נביא את המקור ונבחן מה הצריך שינוי והתאמה. בחינה שכזו תעזור לכם בבואכם להחליף ולהתאים שאלות ישנות כך שיתאימו ליחידה. **שאלות נוספות מסגנון זה הכוללות ניתוח אי ההתאמה ואופן הניסוח המחודש יופיעו בהדרגה במאגר השאלות וכדאי לעקוב**.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

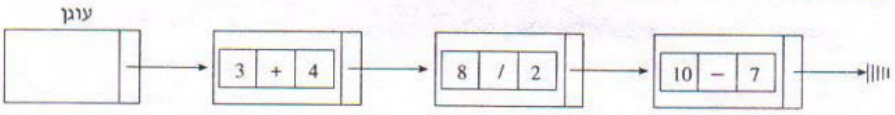
נוסח שאלת הבחינה המקורית:

2. רשימה "חשבונית" L היא רשימה ששדה התוכן שלה מייצג ביטוי חשבוני.

הביטוי החשבוני מורכב משלושה חלקים:

- מספר שלם גדול מאפס
- תו אחד מבין ארבעת התווים:
  - \+ המייצג חיבור
  - \- המייצג חיסור
  - * המייצג כפל
  - / המייצג חילוק
- מספר שלם גדול מאפס

דוגמה לרשימה "חשבונית" L:



א. הגדר בסביבת העבודה את טיפוס שדה התוכן של איבר ברשימה "חשבונית" L.

ב. ממש בסביבת העבודה תת־תכנית calculate, שתקבל רשימה "חשבונית" L ומקום p ברשימה. p הוא מקום ב־ L שאינו סוף־רשימה ואינו עוגן־רשימה. תת־תכנית תחזיר את התוצאה המתקבלת מהביטוי החשבוני הנמצא באיבר שבמקום p.

ג. ממש בסביבת העבודה תת־תכנית sumExpressions, שתקבל רשימה "חשבונית" L, ותחזיר את הסכום הכולל של תוצאות הביטויים החשבוניים הנמצאים ברשימה. בעבור רשימה ריקה יוחזר 0.
עליך להשתמש בתת־תכנית calculate.

בעבור הרשימה "החשבונית" L בדוגמה הנתונה, תת־תכנית sumExpressions תחזיר 14.

הערה: אין צורך לממש בסביבת העבודה את הפעולות של ממשק רשימה.

/המשך בעמוד 5

ניתוח הנוסח המקורי:

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים

הנוסח המקורי מבקש למעשה הגדרת פעולות חיצוניות (תת תוכנית... שתקבל רשימה...). זוהי תוצאה של התכנות הפרוצדורלי שבו לא היו עצמים מטיפוסים שונים. כאשר ממדלים את הבעיה המתוארת בשאלה, בגישה מבוססת עצמים, מגלים שיש צורך להגדיר שני טיפוסים: 'ביטוי חשבוני' שהוא למעשה "טיפוס שדה התוכן" המתבקש בסעיף א. מכיון ש'ביטוי חשבוני' הוא עצם אזי הפעולה המתוארת בסעיף ב, היא אחת מפעולותיו של טיפוס זה.

הטיפוס הנוסף אותו צריך להגדיר הוא 'רשימה חשבונית' שאיבריו הם ביטויים חשבוניים. מכיון שבגישה מבוססת עצמים מדובר בשאלה זו על שתי מחלקות שונות, יש מקום לברר האם התלמיד יכול למדל נכון ולשייך את הפעולות למחלקות המתאימות. (יש לשים לב ששאלת מידול פתוחה, היא שאלה קשה מדי לצורך שאלת מבחן, אך שאלת מידול המשייכת פעולות למחלקות מוגדרות, היא שאלה סבירה בהחלט). תוספת נימוק של התלמיד תעזור לנו לגלות האם הבין מהי חלוקת התפקידים בין המחלקות.

המושג של 'הגדר בסביבת העבודה' מכוון למעשה למתן הייצוג שבו בחר התלמיד עבור המחלקה. ולכן הנוסח הסופי המתקבל כשממירים שאלה זו לנוסחי היחידה החדשים, הוא הנוסח המצוי בספר לתלמיד.

© כל הזכויות שמורות למשרד החינוך ולאוניברסיטה העברית בירושלים