תכנית לימודים במודול

מבוא למדעי המחשב

במקצוע מדעי המחשב לחטיבת הביניים

כיתות ח'-ט'

אלגוריתמיקה – הרחבה והעמקה

תכנית לימודים לכיתה ח' במקצוע מדעי המחשב

כללי:

בהתאם לגישת ההוראה הספירלית, יחידה זו מעמיקה במושגים שכבר נלמדו, ברמה שתואמת את גיל התלמידים ויכולותיהם הקוגניטיביות, ומרחיבה את אוסף המושגים הנלמדים.

הפרדיגמה הנמצאת בבסיסה של היחידה הזאת היא הפרדיגמה האימפרטיבית/פרוצדורלית, ובכך מתאפשרת הרחבה והעמקה באלגוריתמיקה. למרות שהיחידה הקודמת נלמדת באמצעות הפרדיגמה המונעת-אירועים, יש בה חשיפה גם לפרדיגמה האימפרטיבית, שמתבטאת בתכנים (האלגוריתמיים והתכנותיים) בתוך תסריטים שונים של דמויות. התלמידים נחשפו גם לפן הפרוצדורלי דרך ההיכרות הבסיסית עם הפשטה פרוצדורלית. בכל זאת, החשיבה המונעת-אירועים הכתיבה פרקטיקה מסוימת לפתרון בעיות שבה יחידת ההפשטה הבסיסית היא תסריט. לכן, ביחידה הנוכחית התלמידים נחשפים לפרדיגמה שונה לפתרון בעיות, שבה יחידת ההפשטה הבסיסית היא פרוצדורה. לכאורה, שינוי הפרדיגמה מוסיף עומס קוגניטיבי. אבל כאמור, הפער הקונספטואלי במקרה זה אינו מאוד רחב.

למרות שהמעבר דורש למידה נוספת הוא מאפשר להדגים, לחדד ולהעמיק את הרעיונות שהתלמידים נחשפו אליהם או למדו ביחידה הקודמת, ולכן הוא מקדם למידה משמעותית. כל זה נתמך הן על ידי הדמיון בין הפרדיגמות והן על ידי השוני ביניהן. מנגנוני ההפשטה השונים, ובהתאם להם גם תהליכי פתרון הבעיות השונים, מביאים לחשיפה לרעיון הכללי של הפשטה בשני הקשרים שונים, וכך גם לרעיונות הספציפיים יותר של פירוק הדרגתי והסתרת מימוש. חשיפה כזאת מעמיקה את ההבנה של הרעיונות האלו. לעומת זאת, המבנים האלגוריתמים, מבני הנתונים והתבניות האלגוריתמיות רלבנטיים, כאמור, בשתי הפרדיגמות. חשיפה חוזרת אליהם בהקשר של פרדיגמה נוספת מעמיקה את ההיכרות איתם וגם מדגימה את האוניברסליות שלהם ושל רעיון האלגוריתם.

כמובן, כך התלמידים גם נחשפים לרעיון היסודי של פרדיגמה לפתרון בעיות ולהיבטים השונים שלו (למשל, מקביליות היא אינהרנטית לפרדיגמה המונעת-אירועים – משום שאירוע מסוים יכול להביא לביצוע יותר מתסריט אחד, אצל דמות אחת או יותר – אך אינה הכרחית בפרדיגמה האימפרטיבית).

הרחבת הספירלה צריכה להיעשות באופן מבוקר ומוגבל, כדי לעמוד ביעדים של למידה משמעותית. תוספת של מושגים רבים מדי עלולה להביא (משיקולי זמן) להוראה ממוקדת-תיכנות שלהם מחד, ולפחות העמקה במושגים וברעיונות שנלמדו בגיל צעיר, מאידך. ביחידה הזאת הרחבת הספירלה כוללת מבנה נתונים נוסף, תבניות אלגוריתמיות נוספות ופן המימוש של הפשטת נתונים.

היישום כולל מעבר לשפה טקסטואלית (באנגלית) שאינה מבוססת-בלוקים. הוא נעשה בשפה Python, שהתחביר שלה הוא יחסית נקי ופשוט. בהתאם לקווים המנחים האלה חשוב להימנע מהצגת הפגישה המחודשת עם המושגים האלגוריתמיים כהוראה של מילון אשר מראה כיצד מבוטאים המושגים האלה בשפה החדשה. החלק המשמעותי בפגישה המחודשת לא יהיה תרגום סינטקטי משפה לשפה אלא שימוש במושגים האלגוריתמיים כדי לפתור בעיות קשות יותר דרך פתרונות מורכבים יותר, ברמה מותאמת-גיל. בנוסף להעמקה שמושגת כך, לעיתים הפגישה המחודשת מאפשרת הרחבה, משום ההבדלים בין הסביבות (למשל, יחד עם נושא המשתנים יש להתייחס לנושא של טיפוסי נתונים אשר מאוד מובלע בסביבת Scratch).

כדי להקל על העומס הקוגניטיבי במעבר בין שתי הפרדיגמות מומלץ להיעזר בשבועות הראשונים בגישה של mediated transfer אשר נשענת על הקבלה בין פתרונות משתי הפרדיגמות (בדגש קונספטואלי ולא סינטקטי).

שפת Python היא גם שפה מונחית עצמים. בכך תתאפשר חשיפה ראשונית מאוד של התלמידים לעולם מונחה עצמים. אמנם, ביחידה הזאת השימוש בה הוא כשפה פרוצדורלית בעיקרה, אבל התלמידים נחשפים לטיפוסי נתונים שהם למעשה מחלקות בשפה, כמו מערך דינמי ומחרוזת. ניתן לנצל זאת (בהתאם לגישה הספירלית) לזריעה זהירה של זרעים שיהוו בסיס ללמידה מאוחרת יותר של הפרדיגמה המונחית עצמים.

גם בהקשר של היחידה הזאת תקפים הקווים המנחים שהוצגו עבור היחידה הראשונה: גישה מבוססת רמות הפשטה לפתרון בעיות, למידה בהקשר ומבוססת עשייה, אבחנה בין הרבדים – הרובד המושגי, הרובד התכנותי והרובד הטכני, שימוש באמצעים חווייתיים, ובאמצעים שהם unplugged. בנוסף לשימוש בשפה הבסיסית של Python אפשר להשתמש גם בהרחבות של השפה על ידי חבילות גרפיות שיעשירו את אוסף הבעיות המושכות ומעוררות מוטיבציה ועניין. בתכנית שלהלן יש הפרדה בין שעות עיוניות לשעות התנסות, אבל למעשה הן משולבות זו בזו, משום בלמידה בהקשר ומבוססת עשייה מושגים חדשים נלמדים תוך כדי פתרון בעיות, וגם כאשר ניתנות בעיות לתרגול פתרונן יתחיל בחשיבה אלגוריתמית ורק אחר כך יתבצע התכנות

יעדי הלמידה

1. התלמידים יכירו את הרעיון היסודי של פרדיגמה לפתרון בעיות.
2. התלמידים יבינו את ההליך של פתרון בעיות בגישה אימפרטיבית/פרוצדורלית.
3. התלמידים יעמיקו את היכרותם עם הרעיונות היסודיים של מדעי המחשב שהכירו ביחידה הראשונה: הפשטה, פירוק הדרגתי של בעיות, אלגוריתמיזציה, הכמסה.
4. התלמידים יעמיקו את ההיכרות עם המושגים האלגוריתמיים שלמדו בהקשר של הפרדיגמה המונעת-אירועים: נכונות של אלגוריתם, ביצוע סדרתי, מבני בקרה – ביצוע מותנה (פשוט ומקונן) וביצוע חוזר (על תחום סדור ומותנה, פשוט ומקונן), הפשטה פרוצדורלית.
5. התלמידים יעמיקו את היכרותם עם מבנה הנתונים של רשימה (מערך דינמי), אשר כבר פגשו ביחידה הקודמת, ויכירו מבנה נתונים בסיסי נוסף: תור.
6. התלמידים יעמיקו את היכרותם עם הרעיון של הפשטת נתונים וייחשפו גם לפן של מימוש מבנה נתונים.
7. התלמידים יעמיקו בתבניות אלגוריתמיות שכבר פגשו ביחידה הקודמת (מנייה, צבירה, חיפוש פשוט ברשימה) יכירו תבניות אלגוריתמיות נוספות: סכימה, מציאת מינימום, מיון (פשוט), חיפוש (סדרתי) ברשימה ממויינת.
8. התלמידים יידעו ליישם את ההליך של פתרון בעיות בפרדיגמה הפרוצדורלית תוך שימוש במושגים וברעיונות שנלמדו כדי לפתור בעיות אלגוריתמיות ברמות משתנות.
9. התלמידים יערכו היכרות ראשונית מאוד עם רעיונות בסיסיים של העולם המונחה עצמים, דרך השימוש שלהם במחרוזות ורשימות, שהן למעשה עצמים שנושאים עימם שילוב מוכמס של נתונים ופעולות על הנתונים.

נושאי המודול וחלוקת השעות

פרק מס׳	שם הפרק	שעות התנסות	שעות עיוניות	סה״כ שעות
1	**פרק 1: מפגש מחודש עם מושגים אלגוריתמיים בסיסיים דרך פתרון בעיות** **מושגים של מדעי המחשב:** ביצוע סדרתי. משתנים. טיפוסי נתונים. אתחול. פעולות על נתונים. קלט ופלט. עמידות וקריאות. **מושגים תכנותיים של Python:** הגדרה ושימוש במשתנים. אופרטורים חשבוניים. הדפסה למסך. הערות. הזחה. קליטת ערכים מהמקלדת.	5	5	10
2	**פרק 2: מפגש מחודש עם מבנים אלגוריתמיים בסיסיים דרך פתרון בעיות** **מושגים של מדעי המחשב:** טיפוס נתונים בוליאני. תנאים בוליאניים פשוטים ומורכבים. ביצוע מותנה. **מושגים תכנותיים של Python:** פעולות השוואה, אופרטורים בוליאניים. הוראת ביצוע מותנה מתגלגל.	10	10	20
3	**פרק 3: מחרוזות – היכרות ראשונית ומפגש מחודש עם ביצוע חוזר** **מושגים של מדעי המחשב:** מחרוזות כטיפוס נתונים מורכב. התייחסות ראשונית אל מחרוזת כישות שנושאת איתה תוכן ופעולות שניתן לבצע עליו. ביצוע חוזר על פני תחום ערכים סדור מוגדר. **מושגים תכנותיים של Python:** מחרוזות. פעולות על מחרוזות. תחום ערכים סדור (range), הוראת ביצוע חוזר על תחום ערכים סדור מוגדר (for).	15	15	30
4	**פרק 4: מפגש מחודש עם רשימה וביצוע חוזר מותנה** **מושגים של מדעי המחשב:** רשימה (מערך דינמי) כמבנה נתונים מופשט (שימוש בלבד). התייחסות ראשונית אל רשימה כיישות שנושאת איתה תוכן ופעולות שניתן לבצע עליו. ביצוע חוזר מותנה. תבנית סכימה. **מושגים תכנותיים של Python:** הוראת ביצוע חוזר מותנה. רשימה ופעולות על רשימה .	8	7	15
5	**פרק 5: מפגש מחודש עם הפשטה פרוצדורלית** **מושגים של מדעי המחשב:** פונקציות, פרמטרים, ערך מוחזר.	5	5	10

	רעיונות של הפרדיגמה הפרוצדורלית :הפשטה פרוצדורלית. ממשק ומימוש – ממשק כחוזה בין משתמש למממש. הכמסה פרוצדורלית. **מושגים תכנותיים של Python:** פונקציות – הגדרה ושימוש.			
6	**פרק 6: רשימה והפשטה פרוצדורלית - העמקה** **מושגים של מדעי המחשב:** פרמטרים מורכבים. תבנית מציאת מינימום, תבנית מיון פשוט. תבנית חיפוש (סדרתי) ברשימה ממוינת. יעילות – מפגש ראשוני (אינטואיטיבי). **מושגים תכנותיים של Python:** רשימה כפרמטר. פעולת העתקת רשימה.	8	7	15
7	**פרק 7: תור – מבנה נתונים מופשט** **מושגים של מדעי המחשב:** הפשטת נתונים. ממשק. מימוש. הכמסה (בהקשר של הפשטת נתונים). תור כמבנה נתונים מופשט. מימוש תור על ידי רשימה (מערך דינמי).	10	10	20
סה"כ		61	59	**120**

מבוא לאלגוריתמיקה

פרק 1: מפגש מחודש עם מושגים אלגוריתמיים בסיסיים דרך פתרון בעיות

מטרת הפרק

- העמקת ההיכרות עם מושגים אלגוריתמיים בסיסיים שנלמדו בהקשר של הפרדיגמה המונעת-אירועים.
- הכרת אופן המימוש של המושגים האלגוריתמים האלו בשפת Python.
- היכרות מחודשת (ברמה תואמת גיל) עם הרעיון היסודי של פרדיגמה לפתרון בעיות.
- הכרת סביבת העבודה.

מטרות ביצועיות:

- מדעי המחשב

1. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי הכולל שימוש במשתנים.
2. התלמיד יפתח בגישה מודולרית פתרונות אלגוריתמיים המשתמשים בפעולות בסיסיות על מספרים (השוואה, חיבור, חיסור, כפל, חילוק בממשיים ובשלמים).
3. התלמיד יסביר מה הוא טיפוס נתונים ובפרט מה הם הטיפוסים מספר שלם, מספר ממשי ומחרוזת (מחרוזת כטיפוס אטומי פשוט כי בשלב הזה השימוש במחרוזות הוא רק לצורך הדפסה ואין שימוש בפעולות על מחרוזות חוץ מחיבור (+)).
4. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי הכולל שימוש בהוראות קלט דרך הקלדת נתונים והוראות פלט, ומערב את טיפוסי הנתונים שלם, ממשי ומחרוזת

- תכנות בסביבת Python

5. התלמיד יגדיר וישתמש במשתנים בתוך תכניות Python.
6. התלמיד יבחין בין טיפוסים של משתנים .
7. התלמיד יממש ב-Python פתרונות אלגוריתמיים המשלבים פעולות בסיסיות על מספרים שלמים ומספרים ממשיים.
8. התלמיד ישתמש בתוך תכניות Python בהוראות קלט ופלט לצורך קליטה והדפסה של ערכים מספרים ושל תווים או מחרוזות.
9. התלמיד יכלול בתוכנית Python הערות לצורך תיעוד.

- פעולות טכניות בסביבת Python

10. התלמיד יפעיל את סביבת העבודה (כולל שמירה ופתיחת קבצים).
11. התלמיד יכתוב תכניות בסביבת העבודה.
12. התלמיד יריץ תכניות בסביבת העבודה.
13. התלמיד יתקין את סביבת העבודה.

מושגים של מדעי המחשב: ביצוע סדרתי. משתנים. טיפוסי נתונים. אתחול. פעולות על נתונים. קלט ופלט. עמידות וקריאות.

מושגים תכנותיים של Python: הגדרה ושימוש במשתנים. פעולות חשבוניות. הדפסה למסך. הערות. הזחה. קליטת ערכים מהמקלדת.

דרכי ההוראה:

פרק זה, למרות שהוא בסיסי ומערב מושגים פשוטים, הוא מאתגר. מצד אחד, מרבית המושגים האלגוריתמיים הכלולים בפרק הזה כבר מוכרים לתלמידים, מאוד או במידה מסוימת. לכן, בהתאם לגישה הספירלית, הוראת הפרק צריכה להתבסס על הידע הקודם של התלמידים, ובוודאי לא ללמד מחדש את המושגים המוכרים. מצד שני, אמנם את המושגים האלגוריתמיים האלו התלמידים למדו בהקשר של שפה אחת ובקורס הזה הם ישתמשו בהם בהקשר של שפה אחרת, אך לא נרצה שהוראת הפרק תתבסס על תרגום של הוראה משפה אחת להוראה משפה אחרת. זוהי גישה שמתמקדת בתכנות ובעיקר בסינטקס של שפה. בנוסף, גם ביחידה זאת ההוראה והלמידה צריכות להתבצע דרך פתרון בעיות.

יתרה מזאת, הקושי העיקרי אינו במעבר בין השפות, אלא במעבר דרך פרדיגמות (מהפרדיגמה מונעת-האירועים לפרדיגמה הפרוצדורלית), והקושי הזה בא לידי ביטוי בעיקר בפרק הזה. הגישה המתאימה היא גישה שנקראת "העברה מתווכת" (mediated transfer). תוצג בעיה, והיא תיפתר בגישה המוכרת לתלמידים, דרך Scratch, ובגישה החדשה. אין הכוונה להקבלה סינטקטית, אלא קונספטואלית.

בגלל המעבר בין הפרדיגמות בחירת הבעיה המתאימה להעברה המתווכת היא מאתגרת. גם הפרויקטים הבסיסיים ביותר שיצרו התלמידים ב-Scratch כללו לפחות תסריט אחד, והביצוע שלו היה תמיד תלוי בביצוע אירוע (למשל, לחיצה על הדגל הירוק) שהתסריט הנכתב התייחס אליו במפורש. בעיה מתאימה לפתיחת הפרק צריכה להתאים לפרדיגמה הפרוצדורלית, ופתרונה לא יכול לערב אירועים ובוודאי לא מקביליות וריבוי דמויות, כפי שהתרגלו התלמידים ביחידה הקודמת. לכן, הבעיה צריכה להתמקד בהיבט שניתן לפתרון בשתי הפרדיגמות, וימומש דרך קטע של תסריט בודד ב-Scratch ותכנית קצרה ב-Python.

עוד חשוב לשים לב שחלק מהמושגים האלגוריתמיים (בעיה, אלגוריתם, ביצוע סדרתי, אתחול ותיעוד) מוכרים היטב לתלמידים ומשמעותם זהה גם בפרדיגמה החדשה. לכן, ההעברה שלהם היא פשוטה למדי. לעומת זאת המושגים של משתנה והוראות קלט/פלט אמנם מוכרים לתלמידים אך השימוש בהם בהקשר של היחידה הזאת שונה למדי. לגבי משתנים, ב-Scratch אין בכלל טיפוסי נתונים, וזהו מושג חדש עבור התלמידים. אופי העבודה עם משתנים הוא שונה כאשר יש להתייחס לטיפוסי הנתונים. שינוי משמעותי יותר מתייחס להוראות קלט/פלט. יש הוראות כאלו ב-Scratch אך השימוש בהן הוא מוגבל ומנוון למדי. עיקר הקלט ב-Scratch מגיע דרך העכבר או לחיצות על מקשים והטיפול בו נעשה דרך אירועים. ההבדל הזה נעוץ בהבדל הפרדיגמות. בשפה מונעת-אירועים קלט חיצוני הוא בדרך כלל אירוע, כלומר, באחריות מפתח הפרויקט לדאוג לטיפול נכון באירוע כאשר יקרה, אך אין לו שליטה מתי יקרה. לעומת זאת בפרדיגמה הפרוצדורלית מפתח הפרויקט קובע מתי צריך להגיע קלט. ההעברה המתווכת תצטרך להתייחס להבדלים האלו.

ולבסוף, ביחידה הזאת, בהתאם לגישה הספירלית, התלמידים יתמודדו עם בעיות אלגוריתמיות מורכבות יותר מאלו שאיתן התמודדו בכיתה ז'. אבל הבעיה הפותחת צריכה להיות פשוטה כדי שלא ייווצר עומס קוגניטיבי מעבר לזה המתחייב מהשינויים הקונספטואליים הכרוכים במעבר בין הפרדיגמות.

משום החשיבות של הנאמר לעיל, דרכי ההוראה של פרק זה יתוארו בפירוט רב ויציגו יישום אפשרי של הגישה.

1. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג, גם הפשוט ביותר, צריך להילמד בהקשר של פתרון בעיה מסוימת, בעקבות צורך שמתעורר במהלך פתרון הבעיה.
2. בהתאם לכך, ההוראה תתחיל בהצגת בעיה אלגוריתמית פשוטה שניתנת לפתרון בעזרת ביצוע סדרתי תוך שימוש במשתנים מספריים ופעולות עליהם (פעולות על שלמים ועל ממשיים):

- תוך כדי דיון כיתתי התלמידים יפתחו פתרון אלגוריתמי לבעיה. תהליך הפתרון יתמקד בחלק המרכזי בלבד, ללא קלט ופלט.
- התלמידים ידונו בפתרון ויסמנו בו חלקים: אתחול, חישובים, שימוש במשתנים.

3. מימוש הפתרון:

- יוצג (או יפותח ע״י התלמידים) מימוש של הפתרון ע״י קטע של תסריט ב-Scratch.
- יפותח בהדרגה פתרון ראשוני ב-Python. חשוב: הפתרון לא יפותח על בסיס קטע התסריט, אלא ישירות מהאלגוריתם. כל צעד באלגוריתם יתורגם להוראה ב-Python אשר תוסבר תוך כדי הפתרון.
- תתבצע השוואה בין הפתרונות, אשר תתבסס על הרכיבים האלגוריתמיים שזוהו אחרי פיתוח האלגוריתם. ההשוואה לא תהיה רק סינטקטית אלא קונספטואלית. למשל, בפתרון ב-Scratch האתחול של המשתנה מתבצע באופן שונה מעדכון של ערך המשתנה על בסיס ערכו הקיים, בעוד שבפתרון ב-Python אין אבחנה ושתי הפעולות ממומשות על ידי הוראת השמה.
- השוואה נוספת תתבסס על החישובים המתבצעים. ובפרט תתייחס לסוג שלהם. בדיון יעלה שב-Scratch אין צורך לחשוב על סוג המשתנה והכל יעבוד בסדר. דיון מקביל לגבי Python יביא להיכרות ראשונה עם המושג של טיפוסי נתונים, ולדיונים על משמעות הפעולות החשבוניות כאשר יש חשיבות לטיפוס הנתונים (שלם מול ממשי).
- יתקיים דיון על הרצת הפתרון. בפרט, תהיה התייחסות לכך שמימוש הפתרון ב-Scratch אינו מלא משום שאינו יוצר תסריט שלם, והתייחסות לגורמים שמפעילים תסריטים ב-Scratch (לחיצה על הדגל הירוק, קבלת מסר, לחיצה על מקש וכו׳). הדיון יוביל לכך שהגישה שבה יעבדו השנה היא שונה, ואינה תלויה בדברים שמתרחשים מחוץ לתסריטים. אין צורך להתייחס במפורש למילה ׳פרדיגמה׳, אבל חשוב להדגיש את האופי של Scratch כשפה שמבוססת על אירועים, בעוד ש-Python אינה כזאת.
- בנוסף הדיון בהרצת הפתרון יתמקד בתוצאה שלו, כלומר מה יתרחש כשנריץ ומה נוכל לראות. הדיון יוביל לצורך בפקודות פלט.

4. הבעיה הקודמת תורחב כך שיהיה צורך בפלט (פלט מספרי פשוט, ללא מילים). הפתרון האלגוריתמי יורחב בהתאם.
5. יתקיים דיון על מימוש ההרחבה בכל אחת מהשפות:
   - מהן האפשרויות שיש לפלט ב-Scratch (בועות דיבור) ומהן ההוראות המתאימות בשפה?
   - הוראת הפלט שבאלגוריתם תתורגם ל-Python ותיערך השוואה בין שני הפתרונות.
6. הבעיה תורחב כך שיידרש פלט מורכב, המערב מילים ומספרים, והוראת הפלט באלגוריתם תעודכן בהתאם. יתקיים דיון על המימוש בשתי השפות:
   - שימוש בפעולה join ב-Scratch והתייחסות לכך שבעצם מצורפים מילים וערך מספרי אבל אין לכך חשיבות בשפה.
   - התייחסות לכך שמאחר שב-Python יש חשיבות לסוג הנתונים ומאחר שמילים אינן משתנה שלם הן מתאימות לטיפוס נתונים חדש.
7. דיון במענה המתאים לטיפוס הנתונים המתאים. הצגת המושג **האלגוריתמי** של מחרוזת, כסדרה המורכבת מאותיות, מספרים, או כל סימן שעל המקלדת, והתאמת הוראת הפלט שבאלגוריתם.
8. דיון באופי המימוש של הוראת הפלט מהאלגוריתם ב-Python ובתפקיד המחרוזת בהוראה. ניתן לדון בכמה אפשרויות:
   - שימוש בהסבת שלם או ממשי למחרוזת ובחיבור מחרוזות. עריכת השוואה בין המימוש הזה לפעולת join ב-Scratch. שימו לב – שימוש בהסבת ערך מספרי למחרוזת הוא למעשה שימוש בפונקציה. זו אם כך ההיכרות הראשונה עם כתיבה פונקציונלית – שימוש בפונקציה קיימת ע״י העברת ערך וקבלת ערך מוחזר. אין צורך להשתמש בשלב זה במונח ׳פונקציה׳, והמונח ׳הסבה׳ יספיק. מתאים להתייחס בשלב זה אל השימוש בהסבה כאל שימוש במכשיר סגור. איננו יודעים איך הוא עובד, אנחנו רק יודעים איך להפעיל אותו ולהשתמש במה שהוא עושה עבורנו.

- עיצוב מחרוזת הפלט כך שישולבו בה ערכים מספריים.

9. הרחבת הבעיה כך שיידרש גם קלט. הפתרון האלגוריתמי יורחב בהתאם.
10. יתקיים דיון על המימוש של תוספת טיפול בקלט לאלגוריתם בשתי השפות:
    - מהו סוג הקלט המתאים ב-Scratch (קליטת ערך דרך המשתנה "תשובה") ומהן ההוראות המתאימות בשפה?
    - הוראת הקלט שבאלגוריתם תתורגם ל-Python ותיערך השוואה בין שני הפתרונות. בפרט יידונו אופי השימוש במשתנים לצורך הקליטה, וההיבט של טיפוסי הנתונים, כלומר, טיפוס הערך הנקלט ואופי השימוש בו לצורך חישובים מספריים.
    - תתבצע הרצה של הפתרון בסביבת העבודה ב-Python.
11. בהמשך הפרק אין יותר צורך בהעברה מתווכת. בשלב הזה ניתן להציג לתלמידים בעיות מורכבות יותר (ולהעלות בהדרגה את דרגת הקושי) המשתמשות במושגים של ביצוע סדרתי, משתנים, טיפוסי הנתונים שלם, ממשי ומחרוזת, אתחול, פעולות חשבוניות, קלט ופלט. יפותחו פתרונות אלגוריתמיים והם ימומשו ב-Python ויופעלו בסביבת העבודה. חשוב מהשלב הזה והלאה להקפיד על תיעוד ולהדגיש את תפקידו ביצירת תכנית קריאה ועמידה לשינויים.
12. תחילת התהליך מאופיין בשילוב בין מעבדה לבין הוראה עיונית. אין לייחס לסביבת העבודה עצמה חשיבות מיוחדת בתהליך ההוראה ועיקר הדגש הוא על חשיבות האלגוריתם הפשוט ולאחר מכן על מימושו. ההיכרות הראשונית עם סביבת העבודה תתבצע במעבדה לצורך ההרצה של הבעיה הבסיסית (השלבים האחרונים בסעיף 3) ותורחב לצורך הרצת הבעיה המורחבת (הסעיף האחרון בסעיף 10) שמערבת גם קלט ופלט. בעת ההיכרות עם סביבת העבודה יש להדגיש את ההקפדה על הזחה. בשלב הזה אין לעניין ההזחה משמעות אלגוריתמית, והוא מהווה הגבלה טכנית.

<u>דרכי הערכה</u>:

1. פיתוח פתרון אלגוריתמי עבור בעיה נתונה תוך שימוש במשתנים ופעולות עליהם, קלט ופלט.
2. מעקב אחר ביצוע אלגוריתם נתון הכולל ביצוע סדרתי, שימוש במשתנים מטיפוסים שונים, פעולות על מספרים ושימוש בקלט ופלט.
3. משימה תכנותית – כתיבת תכנית לפתרון הבעיה מסעיף 1.
4. מעקב אחרי תכנית נתונה.
5. הסבר (ברמה אלגוריתמית) של תכנית נתונה.

<u>חלוקת השעות</u>

בחלוקה להלן יש הפרדה בין שעות עיוניות לשעות התנסות, אבל כפי שניתן לראות מדרכי ההוראה המוצעות, למעשה הן משולבות זו בזו.

הנושא	**שעות התנסות**	**שעות עיוניות**	**סה"כ שעות**
מושגים אלגוריתמיים	-	5	5
יישום וביצוע בסביבת Python	2	-	2
היכרות עם סביבת העבודה	3	-	3
סה"כ שעות	**5**	**5**	**10**

פרק 2: מפגש מחודש עם מבנים אלגוריתמיים בסיסיים דרך פתרון בעיות

מטרת הפרק

- הכרה עם טיפוס הנתונים הבוליאני, תנאים לוגיים ופעולות בוליאניות
- העמקת ההיכרות עם מבנה הבקרה של ביצוע מותנה

מטרות ביצועיות:

- מדעי המחשב

1. התלמיד יסביר מהו טיפוס הנתונים הבוליאני ובאילו פעולות הוא משמש (כתוצאה או כנתון)
2. התלמיד יפתח בגישה מודולרית פתרונות אלגוריתמיים עבור בעיות ברמת קושי מתאימות הכוללים שימוש בביצועים מותנים עם וללא חלופה, תוך שימוש בתנאים לוגיים ברמות שונות של מורכבות.
3. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי הכולל שימוש בביצוע מותנה מתגלגל.
4. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי הכולל קינון של ביצועים מותנים.

- תכנות בסביבת Python

5. התלמיד יכתוב ביטויים בוליאניים המשתמשים בפעולות השוואה ובפעולות בוליאניות.
6. התלמיד יממש ביצוע מותנה ללא חלופה.
7. התלמיד יממש ביצוע מותנה עם חלופה.
8. התלמיד יממש ביצוע מותנה מתגלגל
9. התלמיד יממש פתרונות אלגוריתמיים לבעיות ע"י תכניות שכוללות ביצועים מותנים מסוגים שונים, שילוב של פעולות השוואה ופעולות בוליאניות בתוך תנאים לוגיים וקינון של ביצועים מותנים.

מושגים של מדעי המחשב: טיפוס נתונים בוליאני. תנאים בוליאניים פשוטים ומורכבים. ביצוע מותנה. בדיקת תקינות קלט.

מושגים תכנותיים של Python: פעולות השוואה, פעולות בוליאניים. הוראת ביצוע מותנה מתגלגל.

דרכי ההוראה:

1. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג עולה תוך כדי פתרון בעיה המשתמש בו. חשוב לבחור בעיות מעניינות. גם בפרק זה מוצגים מושגים המוכרים לתלמידים מהיחידה לכיתה ז', וההוראה מסתמכת על הידע הקודם. עם זאת, הבעיות צריכות להיות בדרגת קושי גבוהה יותר מאלו שנכללו ביחידה הקודמת (אך תואמות גיל). בכך מתפתחת ומעמיקה יכולת פתרון הבעיות של התלמידים.
2. הצגת בעיה ברמת קושי מתאימה המזמנת פתרון אלגוריתמי שכולל ביצוע מותנה ללא חלופה. פיתוח הפתרון האלגוריתמי באופן מודולרי. תוך כדי פיתוח הפתרון יודגשו המושגים האלגוריתמיים המוכרים לתלמיד מהיחידה הקודמת (תנאים לוגיים, פעולות השוואה, פעולות בוליאניות וביצוע מותנה).
3. מימוש הפתרון ב-Python. יש להתייחס בהקשר הזה לטווח של מבנה הבקרה, כלומר מהן הפעולות שהוא כולל. לדון בנקודה זו תוך התייחסות לאופן שבו הטווח מסומן בתיאור האלגוריתמי, לאופן שבו שפת Python מסמנת את הטווח (הזחה), ותוך השוואה לסימון הטווח בלבנה של הביצוע המותנה של Scratch.

4. הצגת (או הרחבת הבעיה הקודמת אל) בעיה המזמנת פתרון אלגוריתמי שכולל ביצוע מותנה עם חלופה. פיתוח הפתרון האלגוריתמי, תוך כדי הדגשת המושג המוכר של ביצוע מותנה עם חלופה, ומימושו ב-Python.
5. הצגת (או הרחבת הבעיה הקודמת אל) בעיה המזמנת פתרון אלגוריתמי שכולל ביצוע מתגלגל. פיתוח הפתרון האלגוריתמי ומימושו ב- Python (בעזרת הוראה מסוג if…elif…else).
6. הצגת (או הרחבת הבעיה הקודמת אל) בעיה המזמנת פתרון אלגוריתמי שכולל קינון של ביצועים מותנים. פיתוח הפתרון האלגוריתמי ומימושו ב-Python.
7. תרגול נוסף של פתרון בעיות מעניינות ברמת קושי מתאימה אשר רלבנטיות למושגים הכלולים בפרק.

<u>דרכי הערכה</u>:

1. מעקב אחר ביצוע אלגוריתם נתון הכולל ביצוע מותנה (בלי/עם חלופה, מתגלגל או מקונן).
2. זיהוי מטרתו של אלגוריתם נתון הכולל בתוכו ביצוע מותנה (עבור סוגי ביצועים מותנים שונים וקינון ביצועים מותנים).
3. פיתוח פתרון אלגוריתמי עבור בעיה נתונה תוך שימוש בביצוע מותנה ללא/עם חלופה.
4. פיתוח פתרון אלגוריתמי עבור בעיה נתונה תוך שימוש בביצוע מותנה מתגלגל.
5. פיתוח פתרון אלגוריתמי עבור בעיה נתונה תוך שימוש בביצוע מותנה מקונן.
6. מימוש פתרון אלגוריתמי המשתמש בביצוע מותנה ללא/עם חלופה.
7. מימוש פתרון אלגוריתמי המשתמש בביצוע מותנה מקונן.
8. בכל המקרים שלעיל יש לשלב תנאים לוגיים ברמות שונות של מורכבות.

<u>חלוקת השעות</u>

הנושא	**שעות התנסות**	**שעות עיוניות**	**סה"כ שעות**
ביטויים לוגיים	3	3	6
ביצוע חוזר מותנה מסוגים שונים	4	4	8
פעולות ותנאים בוליאניים	3	3	6
	10	**10**	**20**

סה"כ שעות

פרק 3: מחרוזות – היכרות ראשונית ומפגש מחודש עם ביצוע חוזר

מטרת הפרק

- היכרות עם מחרוזת כשייכת לטיפוס נתונים מורכב ונושאת איתה תוכן ופעולות שניתן לבצע עליו.
- העמקת ההיכרות עם מבנה הבקרה של ביצוע חוזר מוגבל מראש.

מטרות ביצועיות:

- מדעי המחשב

1. התלמיד יסביר מהי מחרוזת ומהם מאפייניה.
2. התלמיד יסביר את הפעולות בסיסיות על מחרוזת (אורך מחרוזת, גישה למיקום מסוים במחרוזת).
3. התלמיד יפתח בגישה מודולרית פתרונות אלגוריתמיים המשתמשים במחרוזות ובפעולות עליהן (בסיסיות ומורכבות).
4. התלמיד יסביר מהו תחום ערכים סדור.
5. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי הכולל ביצוע חוזר על פני תחום ערכים סדור מוגדר.
6. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשלב ביצוע חוזר על פני תחום ערכים סדור מוגדר ופעולות על מחרוזות.

- תכנות בסביבת Python

7. התלמיד יממש ביצוע חוזר על פני תחום ערכים סדור מוגדר (for).
8. התלמיד יתאר פעולות שונות על מחרוזות ויבחין בין צורת הכתיבה של פעולות כלליות (כגון len), ופעולות המשויכות למחרוזת (כגון lower).
9. התלמיד יממש פתרון אלגוריתמי לבעיה ע"י תכנית המשתמשת בפעולות על מחרוזות.
10. התלמיד יממש פתרון אלגוריתמי לבעיה ע"י תכנית המשתמשת בביצוע חוזר על פני תחום ערכים סדור מוגדר.
11. התלמיד יממש לממש פתרון אלגוריתמי לבעיה ע"י תכנית המשתמשת בשילוב של ביצוע חוזר על פני תחום ערכים סדור מוגדר ופעולות שונות על מחרוזות.

מושגים של מדעי המחשב: מחרוזות כטיפוס נתונים מורכב. התייחסות ראשונית אל מחרוזת כישות שנושאת איתה תוכן ופעולות שניתן לבצע עליו. ביצוע חוזר על פני תחום ערכים סדור מוגדר.

מושגים תכנותיים של Python: מחרוזות. פעולות על מחרוזות. תחום ערכים סדורים (range), הוראת ביצוע חוזר על פני תחום ערכים סדור מוגדר.

דרכי ההוראה:

1. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג, גם הפשוט ביותר, צריך להילמד בהקשר של פתרון בעיה מסוימת, בעקבות צורך שמתעורר במהלך פתרון הבעיה. גם כאן חשוב לבחור בעיות מעניינות, ולהקפיד על הפרדתן מהמושגים ה"מקצועיים". כלומר, להימנע מבעיות העוסקות באופן מפורש במחרוזות (בסגנון "כתוב אלגוריתם המקבל מחרוזת..."), אלא לבחור בבעיות העוסקות במונחים יומיומיים (כגון שמות של אנשים או להקות, האשטאג וכו') שלצורך פתרונן צריך להשתמש במחרוזות.
2. הצגת בעיה המזמנת שימוש פשוט במחרוזות והיכרות ראשונית עם פעולות בסיסיות על מחרוזות (אורך, גישה לתו במקום מסוים).

3. תוך כדי פיתוח הפתרון לדון באופן אינטואיטיבי במחרוזת כטיפוס נתונים מורכב, שנושא אתו תוכן מורכב (סדרה של תווים).
4. מימוש הפתרון ב-Python תוך כדי התייחסות לאופן הכתיבה של הפעולות השונות. הדיון צריך להאיר את אופן הכתיבה של גישה לתו, ולהתמקד באופן הייצוג בשפה של פעולת אורך מחרוזת. זו היכרות שנייה עם כתיבה פונקציונלית (ההיכרות הראשונה היתה בפרק הראשון, דרך ההסבה של ערך מספרי למחרוזת). בשלב הזה כבר אפשר להשתמש במונח 'פונקציה', אך עדיין מתאים להתייחס גם בשלב זה אל השימוש בפונקציה כאל שימוש במכשיר סגור אשר איננו יודעים איך הוא עובד, אנחנו רק יודעים איך להפעיל אותו ולהשתמש במה שהוא עושה עבורנו. בהתאם לעקרונות הגישה הספירלית, חשוב לקשר בין שתי ההיכרויות עם אופן הכתיבה הפונקציונלי. כלומר, להזכיר שכך נכתב גם השימוש בהסבה, ושלמעשה גם ביצוע הסבה הוא שימוש בפונקציה.
5. הרחבת הבעיה כך שפתרונה יזמן גם פעולות נוספות על מחרוזות. אין זה משנה אילו פעולות יהיו אלה, כל עוד אינן מורכבות מדי לתיאור, אבל חשוב שהמימוש שלהן בשפה יהיה כפעולות של המחלקה str. למשל, שינוי כל אותיות המחרוזת לאותיות הקטנות המקבילות, או בדיקה האם כל תווי המחרוזת הן אותיות.
6. מימוש הפתרון ב-Python תוך כדי התייחסות לאופן הכתיבה של הפעולות והשוואת אופן הכתיבה של הפעולות החדשות לאופן הכתיבה של פעולת האורך. זו ההיכרות הראשונה עם כתיבה שמסמנת גישה לפעולה **של** טיפוס. הדיון צריך להאיר את אופייה של מחרוזת בשפה, כשייכת לטיפוס מורכב, שלא רק נושאת איתה תוכן מורכב, אלא משויכות אליה גם פעולות שיכולות לפעול על התוכן שלה. זהו דיון אינטואיטיבי בלבד, שלא מתעלם מאיפיוני כתיבה בשפה שעלולים להעלות סימני שאלה מוצדקים אלא מתמודד איתם ועושה זאת בצורה פשטנית ומותאמת לרמה ולהקשר, ובה בעת גם זורע זרעים ראשונים של ישויות הנושאות איתן תוכן ועצמים. כאשר בהמשך לימודיהם בתיכון יכירו התלמידים עקרונות של תכנות מונחה עצמים, יוכלו לקשר אחורה לכאן ולפרש את זה בהקשר של מחלקות ועצמים. זאת דוגמה ליישום קלאסי של הגישה הספירלית.
7. הצגת בעיה המזמנת פתרון אלגוריתמי שכולל ביצוע חוזר פשוט שיש לבצעו מספר ידוע מראש של פעמים. בשלב הפיתוח האלגוריתמי, כאשר יעלה הצורך במבנה כזה, הפנייה למבנה המוכר לתלמידים מלימודיהם בכיתה ז' – ביצוע חוזר מוגבל מראש – יעלה באופן טבעי.
8. מימוש הפתרון ב-Python. זהו זמן מתאים לשימוש נוסף בפעולת ההעברה המתווכת, כלומר להשוואה של מימוש הביצוע החוזר באלגוריתם ב-Python וב-Scratch. גם הפעם ההשוואה אינה סינטקטית אלא קונספטואלית. הדיון שייערך יכול להוביל להבנה שהמימוש ב-Python למעשה מייצג מבנה בקרה רחב יותר, של ביצוע חוזר על פני תחום ערכים כלשהו שהינו סדור ומורכב מראש.
9. הצגת בעיה המזמנת פתרון אלגוריתמי שכולל ביצוע חוזר פשוט על פני תחום סדור ומוגדר מראש של ערכים מספריים (שאינו מתחיל מ-1). פתרונה ומימוש הפתרון ב-Python בעזרת המבנה החדש.
10. הצגת בעיה המזמנת פתרון אלגוריתמי שמשלב מחרוזות עם ביצוע חוזר על פני תחום ערכים סדור מוגדר . פיתוח הדרגתי של הפתרון האלגוריתמי ומימושו ב-Python.
11. תרגול נוסף של פתרון בעיות המזמנות שילוב דומה, ומימוש הפתרונות בשפה.

<u>דרכי הערכה:</u>

1. מעקב אחר ביצוע אלגוריתם נתון הכולל שימוש במחרוזות ובפעולות על מחרוזות.
2. זיהוי מטרתו של אלגוריתם נתון הכוללים שימוש במחרוזות ובפעולות על מחרוזות.
3. פיתוח של אלגוריתם עבור בעיה נתונה הכולל שימוש במחרוזות ובפעולות על מחרוזות.
4. מימוש של אלגוריתם עבור בעיה נתונה הכולל שימוש במחרוזות ובפעולות על מחרוזות.

5. מעקב אחר ביצוע אלגוריתם נתון הכולל ביצוע חוזר על פני תחום ערכים סדור מוגדר (עבור תחומים שונים).
6. זיהוי מטרתו של אלגוריתם נתון הכולל בתוכו ביצוע חוזר על פני תחום ערכים סדור מוגדר (עבור תחומים שונים).
7. פיתוח של פתרון אלגוריתמי המשתמש בביצוע חוזר על פני תחום ערכים סדור מוגדר.
8. מימוש פתרון אלגוריתמי המשתמש בביצוע חוזר על פני תחום ערכים סדור מוגדר.
9. מעקב אחר ביצוע אלגוריתם נתון הכולל שילוב של מחרוזות עם ביצוע חוזר על פני תחום ערכים סדור מוגדר.
10. זיהוי מטרתו של אלגוריתם נתון הכולל שילוב של מחרוזות עם ביצוע חוזר על פני תחום ערכים סדור מוגדר.
11. פיתוח אלגוריתם עבור בעיה נתונה הכולל שילוב של מחרוזות עם ביצוע חוזר על פני תחום ערכים סדור מוגדר.
12. מימוש פתרון אלגוריתמי הכולל שילוב של מחרוזות עם ביצוע חוזר על פני תחום ערכים סדור מוגדר.

<u>חלוקת השעות</u>

הנושא	**שעות התנסות**	**שעות עיוניות**	**סה"כ שעות**
מחרוזות ופעולות על מחרוזות	5	5	10
ביצוע חוזר על פני תחום ערכים	5	5	10
סדור מוגדר	5	5	10
שילוב של מחרוזות וביצוע חוזר על פני תחום ערכים סדור מוגדר			
סה"כ שעות	**15**	**15**	**30**

פרק 4: מפגש מחודש עם ביצוע חוזר מותנה ורשימה

מטרת הפרק

- היכרות מחודשת עם רשימה (מערך דינמי) כמבנה נתונים מופשט (שימוש בלבד).
- הכרת מחודשת עם מבנה הבקרה של ביצוע חוזר מותנה.

מטרות ביצועיות:

- מדעי המחשב

המטרות הביצועיות בפרק זה, פרט למטרה 5, אינן חדשות וכבר הוגדרו כמטרות הביצועיות של פרק 9 ביחידה לכיתה ז'. עם זאת, מאחר שהבעיות מורכבות יותר, ושינוי הפרדיגמה משמעותי כאן מאוד, רשימת המטרות הביצועיות כוללות גם את המטרות שאינן חדשות.

1. התלמיד יסביר מהי רשימה (מערך דינמי) ומה מטרתה.
2. התלמיד יסביר את הפעולות הבסיסיות על רשימה (יצירת רשימה ריקה, אורך רשימה, גישה למיקום מסוים ברשימה, מחיקת פריט, הוספת פריט בסוף רשימה, הוספת פריט במיקום מסוים ברשימה, עדכון פריט במקום מסוים ברשימה, שרשור רשימות, העתקת רשימה).
3. התלמיד יפתח בגישה מודולרית פתרונות אלגוריתמיים המשתמשים ברשימה ובפעולות על רשימה.
4. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשתמש בביצוע חוזר מותנה.
5. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשתמש בתבנית של סכימת איברים ברשימה.
6. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשלב ביצוע חוזר מותנה ופעולות על רשימות.

- תכנות בסביבת Python

7. התלמיד יממש פתרון אלגוריתמי לבעיה ע"י תכנית המשתמשת ברשימה סדורה ובפעולות סטנדרטיות על רשימה סדורה (אורך רשימה, גישה למיקום מסוים ברשימה, מחיקת פריט, הוספת פריט בסוף רשימה, הוספת פריט במיקום מסוים ברשימה, עדכון פריט במקום מסוים ברשימה, חיבור רשימות, העתקת רשימה).
8. התלמיד יממש פתרון אלגוריתמי הכולל ביצוע חוזר מותנה.
9. התלמיד יממש תבנית סכימה.

מושגים של מדעי המחשב: רשימה (מערך דינמי) כמבנה נתונים מופשט (שימוש בלבד). התייחסות ראשונית אל רשימה כיישות שנושאת איתה תוכן ופעולות שניתן לבצע עליו. ביצוע חוזר מותנה. תבנית סכימה.

מושגים תכנותיים של Python: הוראת ביצוע חוזר מותנה. רשימה ופעולות על רשימה

דרכי ההוראה:

1. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג, גם הפשוט ביותר, צריך להילמד בהקשר של פתרון בעיה מסוימת, בעקבות צורך שמתעורר במהלך פתרון הבעיה. חשוב שהבעיות יהיו מעניינות ובמושגים כלליים (כלומר, לא ינוסחו במונחים של 'רשימות').
2. הצגת בעיה המזמנת שימוש באוסף סדור של ערכים.
3. פיתוח הדרגתי של פתרון אלגוריתמי לבעיה מסעיף 2, שמבצע פעולות על רשימה סדורה ומימושו ב-Python. שימוש בעקרונות של העברה מתווכת כדי לזהות קשרים והבדלים בין מבני הנתונים של

רשימה ב-Scratch וב-Python. בדומה לטיפוס המחרוזת, גם רשימה ב-Python היא טיפוס מורכב הנושא איתו גם תוכן וגם פעולות עליו. לכן צריך לקשר מכאן אל מה שכבר נלמד עבור מחרוזות בפרק הקודם.

4. הרחבת הבעיה באופן הדרגתי כך שתזמן פעולות נוספות על רשימות. לכל אחת מההרחבות יפותח בתורו פתרון אלגוריתמי וימומש ב-Python. כל אחת מהפעולות שיכירו התלמידים תילמד בהקשר של בעיה ולצורך פתרונה. גם כאן, בדומה למחרוזות, יש פעולות שהן למעשה פונקציות כלליות ולא משויכות לרשימה מסוימת (למשל, len או del). לכן חשוב גם כאן לקשר למה שנלמד עבור מחרוזות ולאופן הכתיבה של פנייה לפונקציה כללית ופנייה לפעולה של רשימה.
5. הצגת בעיה המזמנת שימוש בביצוע חוזר מותנה בהקשר של רשימות. פיתוח פתרון אלגוריתמי עבור הבעיה. דיון במבנה של ביצוע חוזר מותנה המוכר לתלמידים מכיתה ז'. מימוש הפתרון ב-Python. המימוש של מבנה הביצוע החוזר המותנה ב-Python דומה מאוד למימוש שלו ב-Scratch ולכן למעשה אין כאן חידוש של ממש.
6. הצגת בעיה המזמנת סכימה של ערכים באוסף סדור של איברים.
7. פיתוח פתרון אלגוריתמי לבעיה מסעיף 6 אשר משתמש בתבנית סכימה על פני רשימה. לקשר אותו לתבנית הצבירה שנלמדה בכיתה ז'. מימוש האלגוריתם החדש ב-Python.

<u>דרכי הערכה:</u>

1. מעקב אחר ביצוע אלגוריתם נתון הכולל שימוש ברשימות.
2. זיהוי מטרתו של אלגוריתם נתון הכולל רשימות.
3. פיתוח ומימוש של פתרון אלגוריתמי עבור בעיה המצריכה שימוש ברשימות.
4. פיתוח ומימוש של פתרון אלגוריתמי עבור בעיה המצריכה שימוש בתבנית סכימה.
5. מעקב אחר אלגוריתם אשר משתמש בהפשטה פרוצדורלית.
6. פיתוח ומימוש של פתרון אלגוריתמי עבור בעיה נתונה, אשר משתמש בהפשטה פרוצדורלית.

<u>חלוקת השעות</u>

הנושא	**שעות התנסות**	**שעות עיוניות**	**סה"כ שעות**
רשימה ופעולות על רשימה	2	4	6
ביצוע חוזר מותנה	2	1	3
סכימת איברים ברשימה	2	1	3
חיפוש סדרתי ברשימה ממוינת	2	1	3
סה"כ שעות	**8**	**7**	**15**

פרק 5: מפגש מחודש עם הפשטה פרוצדורלית

מטרת הפרק

העמקת ההיכרות עם הפשטה פרוצדורלית (הגדרת קופסה אלגוריתמית שחורה שממשת פעולה מופשטת ושימוש בקופסאות אלגוריתמיות שחורות), ההעמקה מתבטאת בדיון מפורש יותר על המשמעות של הפשטה פרוצדורלית ובפתרון בעיות מעט מורכבות יותר. העמקה נוספת תיערך בפרק הבא.

מטרות ביצועיות:

- מדעי המחשב

גם בפרק זה מרבית המטרות הביצועיות (1-3) אינן חדשות וכבר הוגדרו כמטרות הביצועיות של פרק 10 ביחידה לכיתה ז'. עם זאת, מאחר שבפרק זה יש דיון מעמיק יותר בעקרונות של הפשטה פרוצדורלית, ומאחר שהבעיות מורכבות יותר, גם את המטרות שאינן חדשות נכללות ברשימה של הפרק הנוכחי. כמו כן, נוספה מטרה רביעית שמדגישה את העיסוק העמוק יותר. אין מדובר רק בשימוש בפונקציות ובמימוש פונקציות אלא בתהליך הוליסטי של פתרון בעיות. אין צורך להשתמש במילה 'פרוצדורלית' אשר לא שימושית בהקשר של Python, ואפשר להשתמש בתהליך ההוראה במונח 'הפשטה אלגוריתמית', כאשר עולה בכך הצורך.

1. התלמיד יסביר מהי הפשטה פרוצדורלית (אלגוריתמית) ומה מטרתה.
2. התלמיד יפתח בגישה מודולרית פתרונות אלגוריתמיים המשתמשים בהפשטה פרוצדורלית כקופסה שחורה (עם או בלי פרמטרים).
3. התלמיד יכתוב אלגוריתמים למימוש קופסאות שחורות.
4. התלמיד יישם את ההליך של פתרון בעיות בפרדיגמה הפרוצדורלית תוך שימוש במושגים וברעיונות שנלמדו כדי לפתור בעיות אלגוריתמיות ברמות משתנות.

- תכנות בסביבת Python

5. התלמיד יממש הגדרת פונקציה.
6. התלמיד ישתמש בפונקציה שהוגדרה ומומשה על ידו.
7. התלמיד יממש פתרון אלגוריתמי לבעיה ע"י תכנית המשתמשת בהפשטה פרוצדורלית.

מושגים של מדעי המחשב: פונקציות, פרמטרים, ערך מוחזר.

רעיונות של הפרדיגמה הפרוצדורלית: הפשטה פרוצדורלית. ממשק ומימוש – ממשק כחוזה בין משתמש לממשם. הכמסה פרוצדורלית.

מושגים תכנותיים של Python: פונקציות – הגדרה ושימוש.

דרכי ההוראה:

1. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג, גם הפשוט ביותר, צריך להילמד בהקשר של פתרון בעיה מסוימת, בעקבות צורך שמתעורר במהלך פתרון הבעיה. חשוב שהבעיות הנתונות יהיו מעניינות ובמושגים כלליים (כלומר, לא בעיות על 'רשימות', 'מחרוזות', 'פונקציות' וכדומה, אלא שאלו יוכלו לשמש בפתרונן).
2. אין צורך להשתמש במילה 'פרוצדורלית' אשר לא שימושית בהקשר של Python. אפשר להשתמש בתהליך ההוראה במונח 'הפשטה אלגוריתמית', כאשר עולה בכך הצורך.
3. הצגת בעיה המזמנת שימוש בהפשטה אלגוריתמית דרך קופסה שחורה.
4. פיתוח הדרגתי של פתרון אלגוריתמי לבעיה מסעיף 2, אשר משתמש באבן בניין פרוצדורלית. אפשר להשתמש עבורה במונח 'קופסה אלגוריתמית'. כאמור, גם את ההפשטה הפרוצדורלית התלמידים

פגשו בכיתה ז', דרך הגדרת לבני משתמש ב-Scratch. חשוב במיוחד לחזור ולהדגיש כאן את עקרון ההחבאה של פרטי הביצוע – מי שמשתמש בפונקציה החדשה לא יודע מה יש בתוכה. הוא יודע רק מה היא עושה ולא איך היא עושה את זה, ויודע איך להפעיל אותה. לדבר על היתרונות של החבאה של פרטי ביצוע. מצד שני, יש לדון בחלק המשותף, בממשק שבו התפר בין המשתמש למיישם. מה החשיבות של הממשק כהסכם מחייב בין שני צדדים? (המילה 'החבאה' משמעותה בהקשר הזה הוא למעשה 'הכמסה' אבל השימוש בה מתאים יותר לצורך היחידה)

5. מימוש הפתרון האלגוריתמי מסעיף 3 ב-Python. גם במקרה הזה תתאים הוראה מתווכת כדי להשוות בין הפרשנות של מושג ההפשטה האלגוריתמית בין שתי השפות. הבדל משמעותי הוא שלבנת משתמש ב-Scratch לא יכולה להחזיר ערך, שלא כמו פונקציות ב-Python.

<u>דרכי הערכה:</u>

1. מעקב אחר אלגוריתם אשר משתמש בהפשטה פרוצדורלית.
2. פיתוח ומימוש של פתרון אלגוריתמי עבור בעיה נתונה, אשר משתמש בהפשטה פרוצדורלית.

<u>חלוקת השעות</u>

הנושא	**שעות התנסות**	**שעות עיוניות**	**סה"כ שעות**
מימוש פונקציות	5	2	5
החבאה וממשק	-	3	5
סה"כ שעות	**5**	**5**	**10**

פרק 6: רשימה והפשטה פרוצדורלית – העמקה

<u>מטרת הפרק</u>

העמקת היכרות עם רשימה (מערך דינמי) כמבנה נתונים מופשט. העמקה בתבניות אלגוריתמיות נוספות (תבנית מציאת מינימום, מיון פשוט, תבנית חיפוש סדרתי ברשימה ממוינת). חשיפה ראשונית לרעיון של יעילות אלגוריתמית. רשימה כפרמטר.

<u>מטרות ביצועיות:</u>

- <u>מדעי המחשב</u>

1. התלמיד יסביר מהי תבנית מציאת מינימום.
2. התלמיד יסביר מהי תבנית מיון פשוט.
3. התלמיד יסביר מהי תבנית חיפוש סדרתי ברשימה ממוינת.
4. התלמיד יסביר מהו אלגוריתם יעיל ברמה האינטואיטיבית.
5. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשתמש בתבנית מינימום.
6. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשתמש בתבנית מיון פשוט.
7. התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשתמש בתבנית חיפוש סדרתי ברשימה ממוינת.
8. . התלמיד יפתח בגישה מודולרית פתרון אלגוריתמי המשלב הפשטה פרוצדורלית וטיפול ברשימות.

- <u>תכנות בסביבת Python</u>

9. התלמיד יממש תבנית מציאת מינימום.
10. התלמיד יממש תבנית מיון פשוט ברשימה.
11. התלמיד יממש תבנית חיפוש סדרתי ברשימה ממוינת.
12. התלמיד יממש הגדרת פונקציה עם רשימה כפרמטר.
13. התלמיד יממש פתרונות אלגוריתמיים אשר משלבים קופסאות אלגוריתמיות שפועלות על רשימות.

מושגים של מדעי המחשב: פרמטרים מורכבים. תבנית מיון פשוט. תבנית חיפוש (סדרתי) ברשימה ממוינת. יעילות – מפגש ראשוני (אינטואיטיבי).

מושגים תכנותיים של Python: רשימה כפרמטר. העתקת רשימה.

<u>דרכי ההוראה:</u>

1. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג, גם הפשוט ביותר, צריך להילמד בהקשר של פתרון בעיה מסוימת, בעקבות צורך שמתעורר במהלך פתרון הבעיה.
2. הצגת בעיה המזמנת מיון אוסף סדור של איברים.
3. פיתוח פתרון אלגוריתמי לבעיה מסעיף 2. הפתרון יפותח בשלבים. לצורך פתרון תת הבעיה העוסקת במיון עצמו ייערך דיון שבמהלכו יעלו הצעות שונות כיצד למיין סדרה של פריטים. המטרה היא להגיע למיון פשוט, אשר בונה רשימה ממוינת באופן הבא: מוצא כל פעם ברשימה הנתונה את האיבר הקטן ביותר הבא ומכניס אותו לרשימה החדשה.
4. הדיון יתמקד בבעיית מציאת המינימום, כדי שאפשר יהיה למצוא בכל פעם את האיבר הבא ברשימה. יפותח אלגוריתם שמוצא את המספר הקטן ביותר ברשימה.

5. לאחר מכן ימשיך פיתוח הפתרון עבור המיון שתואר בסעיף 3.
6. לאחר מכן ימשיך פיתוח הפתרון לבעיה מסעיף 2.
7. ימומשו האלגוריתמים השונים. בשלב ראשון ימומש האלגוריתם למציאת מינימום.
8. בשלב השני ימומש האלגוריתם למיון המשתמש באלגוריתם למציאת מינימום. המימוש ב-Python ישתמש במימוש מסעיף 7 כפונקציה. תוך כדי הדיון במימוש יתברר הצורך להעביר לפונקציה רשימה כפרמטר, כדי שתוכל למצוא בה את האיבר המינימלי. בהמשך הדיון יתברר שכדי שאפשר יהיה למצוא כל פעם את האיבר המינימלי הבא צריך יהיה למחוק בכל פעם את האיבר המינימלי האחרון שנמצא מתוך הרשימה. תעלה השאלה כיצד אפשר יהיה לשמור על הרשימה המקורית ואז יסתבר שיש צורך בפעולת העתקה של רשימות, כדי להעתיק את הרשימה הנתונה לרשימה אחרת שאותה אפשר יהיה לשנות.
9. לבסוף ימומש האלגוריתם המלא שמשתמש בפעולת המיון כפונקציה.
10. הצגת בעיה המזמנת חיפוש אחר ערך באוסף סדור של איברים.
11. יתקיים דיון בבעיה. ביחידה הקודמת התלמידים כבר למדו לחפש באופן פשוט ערך בתוך רשימה – עוברים על הרשימה מתחילתה עד שמוצאים אותו או עד שמגיעים לסוף הרשימה בלי למצוא אותו. כעת ידונו אם אפשר לעשות את החיפוש בצורה חכמה יותר. כך שאם ערך לא נמצא לא נצטרך לעבור את כל הרשימה עד שנדע שהוא לא ברשימה. הדיון יוביל לכך שאם הרשימה ממוינת לא צריך לעבור על כל הרשימה, כי ברגע שמצאנו איבר שגדול ממנו כבר נדע שהערך לא ברשימה.
12. פיתוח פתרון אלגוריתמי לבעיית החיפוש ברשימה ממוינת. קודם כל יהיה שימוש באלגוריתם למיון שפותח בסעיף 5. ואז פיתוח החלק של החיפוש, שיוצר תבנית לחיפוש מספר ברשימה ממוינת.
13. ייערך דיון אינטואיטיבי שישווה בין שתי צורות החיפוש – מי טובה יותר ולמה. אין הכוונה ללמד את מושג היעילות באופן מעמיק. מספיק רק שהדיון יוביל להבנה שלאותה בעיה יכולים להיות כמה פתרונות, וחלקם טובים יותר מאחרים, בכך שהם עושים פחות עבודה.
14. פיתוח אלגוריתם עבור הבעיה מסעיף 10.
15. מימוש האלגוריתמים שפותחו בסעיפים 12 ו-13.
16. פתרון (פיתוח ומימוש) של בעיות המזמנות שימוש בתבניות שנלמדו – מציאת מינימום, מיון פשוט, חיפוש ברשימה ממוינת.

<u>דרכי הערכה:</u>

1. מעקב אחר ביצוע אלגוריתמים נתונים הכוללים תבנית מיון מציאת מינימום.
2. מעקב אחר ביצוע אלגוריתמים נתונים הכוללים תבנית מיון פשוט.
3. מעקב אחר ביצוע אלגוריתמים נתונים הכוללים תבנית חיפוש סדרתי ברשימה ממוינת.
4. פיתוח ומימוש של פתרון אלגוריתמי שמשתמש בקופסאות אלגוריתמיות עבור בעיה המצריכה שימוש בתבנית מיון.
5. פיתוח ומימוש של פתרון אלגוריתמי שמשתמש בקופסאות אלגוריתמיות עבור בעיה המזמנת חיפוש סדרתי ברשימה ממוינת.
6. פיתוח ומימוש של פתרון אלגוריתמי אשר משלב קופסאות אלגוריתמיות שפועלות על רשימות.

<u>חלוקת השעות</u>

הנושא	**שעות התנסות**	**שעות עיוניות**	**סה"כ שעות**
רשימה כפרמטר	2	1	3
תבנית מציאת מינימום	2	1	3

תבנית מיון פשוט	2	2	4
תבנית חיפוש סדרתי ברשימה ממויינת	2	2	4
יעילות		1	1
סה"כ שעות	**8**	**7**	**15**

פרק 7: תור – מבנה נתונים מופשט

מטרת הפרק

היכרות פשטנית עם מבנה נתונים בסיסי נוסף: תור. היכרות עם הרעיון של הפשטת נתונים והחבאת פרטים בהקשר של מימוש מבנה הנתונים תור.

מטרות ביצועיות:

- מדעי המחשב

כפי שמפורט בהמשך, בדרכי ההעברה, מדובר כאן בהיכרות ראשונית. כמבנה נתונים תור הוא פשוט יותר מרשימה, ואין הכוונה שהמימוש יהיה דרך מחלקה כי בכיתה ח' אין עיסוק בתכנות מונחה עצמים. כפי שניתן לראות בהמשך, המימוש עצמו הוא מאוד פשוט והמטרה החשובה היא להדגיש את עקרון ההפשטה – גם הפשטת נתונים (כי מי שמשתמש בתור לא צריך לדעת איך התור עצמו נשמר) וגם הפשטה אלגוריתמית (כלומר, הפרדה בין שימוש בפעולות של תור לבין מימוש שלהן), אשר מוכרת לתלמידים כבר מכיתה ז'..

1. התלמיד יסביר מהו תור ומהי מטרתו.
2. התלמיד יסביר מהן הפעולות הבסיסיות שניתן לבצע על תור (יצירת תור ריק, הכנסת פריט, הוצאת פריט, הצצה לראש התור) ומה הממשק של כל אחת מהפעולות.
3. התלמיד יסביר מהו מבנה נתונים.
4. התלמיד יממש את הפעולות השונות של תור על פי הממשקים שהוגדרו במטרה 2, בהתאם לרעיון הפשטת נתונים והחבאת פרטים.
5. התלמיד ישתמש בפעולות שפיתח עבור מבנה הנתונים תור.
6. התלמיד יפתח בגישה מודולרית פתרונות אלגוריתמיים המשתמשים במבנה הנתונים תור בצורת הפשטה פרוצדורלית כקופסה שחורה.

- תכנות בסביבת Python

7. התלמיד יממש תור באמצעות רשימה (מערך דינמי).
8. התלמיד ישתמש במבנה הנתונים תור שהוגדר ומומש על ידו.
9. התלמיד יממש פתרון אלגוריתמי לבעיה ע"י תכנית שמשתמשת בתור שמימש.

מושגים של מדעי המחשב: הפשטת נתונים. ממשק. מימוש. הכמסה (בהקשר של הפשטת נתונים). תור כמבנה נתונים מופשט. מימוש תור על ידי רשימה (מערך דינמי).

דרכי ההוראה:

1. בפרק הזה מוצג מושג שלא הוזכר ביחידה הקודמת וזאת ההיכרות הראשונה אתו. לכן המושג הזה מהווה הרחבה של הספירלה. זהו מבנה הנתונים של תור. זהו מבנה נתונים פשוט, אפילו יותר פשוט מאשר רשימה. אי אפשר לגשת לכל פריט בתור, רק לזה שבראש התור. אי אפשר לשנות פריטים בתור, ואי אפשר להכניס פריטים בכל מקום בתור, רק בסופו. התלמידים ישתמשו בתור, כמו שהם משתמשים במחרוזת או רשימה, אבל יכתבו גם פונקציות שממממשות את הפעולות על תור, ולכן דרך מבנה הנתונים הפשוט הזה אפשר להעמיק ולהרחיב בנושא הפשטת הנתונים. שימו לב שמבנה הנתונים החדש לא ימומש בעזרת מחלקה. היחידה הזאת אינה עוסקת בתכנות מונחה עצמים. התור ימומש ע"י רשימה ופעולות על התור יהיו פונקציות. בגלל שהמבנה הוא פשוט קל מאוד לכתוב פונקציות שממממשות את הפעולות האלו. אבל דרכן ניתן יהיה להדגים את החבאת הפרטים - רק הפונקציות האלו יודעות באיזה אופן הנתונים של התור נשמרים. מי שישתמש בתור לא צריך לדעת שהתור נשמר כרשימה.

2. הפרק יילמד תוך ביצוע משימות מובנות ומשימות התנסות חופשית בסביבה. כל מושג, גם הפשוט ביותר, צריך להילמד בהקשר של פתרון בעיה מסוימת, בעקבות צורך שמתעורר במהלך פתרון הבעיה.
3. הצגת בעיה לא מורכבת המזמנת שימוש בתור. חשוב שהשימוש בתור לצורך פתרון הבעיה יהיה טבעי וינבע מתוך הבעיה, אבל גם חשוב שהיא תהיה פשוטה, כדי שאפשר יהיה להתמקד בנושא של הפשטת הנתונים.
4. דיון התחלתי בפתרון הבעיה, והתמקדות בנתונים שאליהם מתייחסת הבעיה. מה מאפיין אותם? במה הם דומים לרשימה או למחרוזת? דרך הדיון מגיעים לכך שהנתונים האלה צריכים להישמר באופן שמדמה תור, כמו תור לכניסה להופעה, ומחדדים איך תור כזה מתנהג ומה אפשר לעשות איתו (להכניס משהו בסוף התור, להוציא מראש התור, לראות מה יש בראש התור).
5. פיתוח הדרגתי של פתרון אלגוריתמי לבעיה מסעיף 4. בהתאם לבעיה שנבחרה, האלגוריתם יהיה פשוט למדי אבל כחלק מתהליך הפיתוח צריך יהיה להתמקד בשימוש בתור ובפעולות עליו. ותוך כדי הפיתוח יתבררו פעולות נוספות שיש בהן צורך, כמו יצירת תור ריק: מאחר שמי שמשתמש בתור לא יודע איך הוא נראה בפנים הוא לא יכול ליצור אותו לבד. הוא צריך להפעיל פעולה שיוצרת תור ריק ונותנת אותו למי שישתמש בו. אם כך, כחלק משלב הפיתוח למעשה הוגדר הממשק של מבנה הנתונים – מהו אוסף הפעולות שבעזרתן אפשר להשתמש בו.
6. הפתרון האלגוריתמי שפותח בסעיף 5 ימומש ע"י קטע תכנית ב-Python. כחלק מתהליך המימוש צריך יהיה לדון ולהגדיר את הממשק של הפעולות – מה תקבל כל פעולה ומה היא תחזיר. פעולת היצירה לא צריכה לקבל כלום אבל צריכה להחזיר תור ריק. פעולות ההכנסה, ההוצאה וההצצה צריכות לקבל את התור שעליו הן אמורות לפעול. פעולת ההכנסה צריכה לקבל גם את הפריט שצריך להכניס לתור. פעולת ההכנסה צריכה להחזיר את התור החדש אחרי שהוכנס הפריט החדש בסופו, פעולת ההצצה צריכה להחזיר את האיבר שבראש התור, ופעולת ההוצאה צריכה להחזיר את התור החדש, אחרי שהוצא ממנו האיבר שבראש התור.
7. אחרי שהממשק של הפעולות ברור אפשר לכתוב את קטע התכנית שמשתמש בפעולות האלו, אבל התכנית לא מוכנה להרצה כי הפונקציות המתאימות לתור עדיין לא נכתבו.
8. השלב הבא הוא לפתח פתרונות אלגוריתמיים עבור הפעולות על תור. האלגוריתמים המתאימים הם מאוד פשוטים, בני שורה אחת. אבל ההתמקדות היא בעניין הפשטת הנתונים והחבאת הפרטים. רק בתוך הקופסאות האלגוריתמיות האלו אפשר לדעת איך תור באמת נראה. כדאי לדון בקשר בין החבאת פרטי ביצוע – שאותה הכירו התלמידים לצורך הפשטה אלגוריתמית – להפשטת נתונים. במקרה הראשונים מוחבאים הפרטים של איך מבצעים משהו ובמקרה השני מוחבאים הפרטים של איך משהו נשמר.
9. מימוש הפתרונות האלגוריתמיים ב-Python, כפונקציות. כאמור, הפונקציות יהיו קצרות מאוד ופשוטות. הנקודה החשובה, שבה צריך להתעמק, היא שהפונקציות האלו מחזירות או מקבלות כפרמטר רשימה, אבל מבחינת מי שמשתמש בהן זה בעצם תור, והוא יכול להשתמש בו רק בעזרת הפעולות שבממשק.

<u>דרכי הערכה:</u>

1. מעקב אחר אלגוריתם אשר משתמש בתור.
2. פיתוח ומימוש הרחבה פשוטה לממשק של תור (למשל, הדפסת תור, או בדיקה האם תור הוא ריק).
3. פיתוח ומימוש פתרון אלגוריתמי עבור בעיה נתונה, המשתמש בממשק של מבנה הנתונים תור.

<u>חלוקת השעות</u>

הנושא	שעות התנסות	שעות עיוניות	סה"כ שעות
תור כמבנה נתונים מופשט ופעולות	4	5	9
על תור	2	2	4
שימוש בפעולות של תור	4	3	7
מימוש של תור והפעולות עליו בעזרת רשימה			
סה"כ שעות	**10**	**10**	**20**