מדעי המחשב

כללי

1. הבחינה במדעי המחשב נמנית עם **בחינות הבחירה המחייבת**.

2. אפשר להיבחן במדעי המחשב בהיקף של 3 יח"ל ובהיקף של 5 יח"ל.

3. הבחינה בהיקף של 3 יח"ל מורכבת מן השאלונים שמספריהם 601 + 602.

4. הבחינה בהיקף של 5 יח"ל מורכבת מן השאלונים שמספריהם 601 + 602+ 603.

5. כדי לקבל ציון סופי במקצוע **מדעי המחשב** יש לקבל 55 נקודות לפחות בכל אחד מן השאלונים 602 ו-603.

6. תכנית הלימודים במדעי המחשב מובאת בדפים שלפניכם, וכן באתרי האינטרנט האלה: אתר מדעי המחשב וטכנולוגיות מידע, שכתובתו היא http://www/csit.org.il, אתר משרד החינוך, אתרי משפחת או"ח ואתר המפמ"ר למדעי המחשב.

7. יש להיכנס לאתרים אלה, ולהתעדכן בתכנית הלימודים.

8. באתרים אלה יש גם דוגמאות של בחינות בגרות.

נושאי הלימוד במדעי המחשב
יחידת לימוד אחת
שאלון מספר 601

כללי

שאלון הבחינה במדעי המחשב בהיקף של יחידת לימוד אחת כולל **שני מסלולי בחירה**, שיש לבחור ב**אחד** מהם:

מסלול בחירה 1 – שפת בייסיק או פסקל

מסלול בחירה 2 – שפת תמלילוגו

הנבחנים בהיקף של יחידת לימוד **אחת** יסיימו את לימודי 'מבוא למדעי המחשב'.
נבחנים שירצו להיבחן בהיקף של 3 יח"ל יהיו רשאים להשתלב בלימודי מדעי המחשב בהיקף של 2 יחידות לימוד נוספות, על-פי תכנית הלימודים המחייבת.

מסלול בחירה 1 – שפת בייסיק או פסקל

בשאלון **שלושה** פרקי חובה – 1, 2 ו-3, ופרק בחירה **אחד** – מבין הפרקים 4 ו-5.

פרק 1: הכרת המחשב

- המחשב מהו: מבנה המחשב – מעבד, זיכרון RAM; אמצעי קלט ופלט – מקלדת ועכבר, צג, מדפסת והתקנים נוספים (מודם, כרטיס קול, צורב, סורק וכדומה)
- אמצעי זיכרון חיצוניים: כונן תקליטונים ותקליטון מגנטי; תקליטון (דיסק) קשיח; כונן תקליטורים ותקליטור (CD-ROM ,DVD)
- הפעלת המחשב ושימוש במערכת ההפעלה: התנסות בפעולות בסיסיות כגון טעינת קובץ, שמירת קובץ, שימוש בתפריטים, העתקת קבצים, שמירה בתקליטון או בתיקייה, מחיקת קבצים
- שימוש בעורך

פרק 2: מושגי יסוד במדעי המחשב

אלגוריתם ותכונותיו, אלגוריתם מילולי ככלי להצגת פתרון לבעיה, תרשים זרימה, אימות האלגוריתם על-ידי שימוש בטבלת מעקב, הצגת פתרון לבעיה נתונה באמצעות עידון הדרגתי

פרק 3: יסודות התכנות א'

התכנית הכללית, הוראות קלט ופלט, סוגי נתונים ומשתנים (כולל מחרוזות), הדפסת כותרות, הוראות הצבה, פעולות אריתמטיות, פונקציות ספרייה: שורש ריבועי, ערך מוחלט, מספרים אקראיים, החלק השלם

פרק 4: יסודות התכנות ב'

- עקרונות התכנות המובנה, שגרה ללא פרמטרים
- מבני בקרה
- לולאות FOR ו-WHILE
- משפטי תנאי IF-THEN-ELSE

פרק 5: כלי תוכנה ויישומיהם

- הכרת הגיליון האלקטרוני – פרק חובה
- הכרת הגיליון ותכונותיו, שימוש בעזרה (Help) של התכנית, הכנסת מידע לגיליון ודפדוף בו
- יישומים חישוביים בגיליון: הכנסת נוסחאות פשוטות – הפקודות COPY ו-MOVE; הפונקציות INT, MIN, MAX, AVG, SUM, COUNT; כתובות יחסיות ומוחלטות (ללא כתובות מעורבות); הוראת IF בסיסית; טעינה ושמירה
- הכרת מסד מידע: הכרת מסד המידע ותכונותיו, הכנסת מידע למסד ודפדוף בו; עדכון מידע, הדפסת דוח, טעינה ושמירה
- הכרת התמלילן: הכרת התמלילן ותכונותיו, כתיבה ועריכה, הדפסה, טעינה ושמירה
- הכרת הסביבה הגרפית: הכרת המחולל הגרפי ותכונותיו, ציור ועריכת סרטוט, הדפסה, טעינה ושמירה

מסלול בחירה 2 – שפת תמלילוגו

פרק 1: הכרת המחשב

- המחשב מהו: מבנה המחשב – מעבד, זיכרון RAM; אמצעי קלט ופלט – מקלדת ועכבר, צג, מדפסת והתקנים נוספים (מודם, כרטיס קול, צורב, סורק וכדומה)
- אמצעי זיכרון חיצוניים: כונן תקליטונים ותקליטון מגנטי; תקליטון (דיסק) קשיח; כונן תקליטורים ותקליטור (DVD, CD-ROM)
- הפעלת המחשב ושימוש במערכת ההפעלה: התנסות בפעולות בסיסיות כגון: טעינת קובץ, שמירת קובץ, שימוש בתפריטים, העתקת קבצים, שמירה בתקליטון או בתיקייה, מחיקת קבצים

פרק 2: מושגי יסוד במדעי המחשב

- אלגוריתם ותכונותיו
- אלגוריתם מילולי ככלי להצגת פתרון לבעיה

פרק 3: יסודות התכנות א'

- היכרות כללית עם הסביבה התכנותית: לוח המקשים, אובייקטים תכנותיים והוראות ראשונות בשפת התכנות; עבודה עם תקליטון, פרויקטים ודפים (קבצים) ומדורים (מחיצות), הפעלת המחשב באופן עצמאי, הכנת תרגיל מסכם
- תכנות בסיסי: הליכים פשוטים והוראת חזרה, שימוש בעורך והכרת התמלילן, כתיבת הליכים (פרוצדורות) חדשים; כתיבת תכניות קצרות תוך שימוש ב-2–3 הליכים; הוראת החזרה "חזור" ומושג הלולאה; אתחול לולאות; הוראת ההדפסה
- מושג התכנית: הבניה, עידון הדרגתי וניפוי שגיאות; התנסות בטעינת תכניות מורכבות ובהרצתן; קריאה והבנה של תכנית הבנויה מתת-הליכים; הכנסת שינויים בתכנית וניפוי שגיאות; היכרות ראשונה עם מושג האלגוריתם; הכנת פרויקט תכנותי מסכם (הנחיות במדריך למורה); הכנת תיק פרויקט והגשתו, כולל הסבר כללי, תדפיס התכנית ותקליטון עם הקבצים הדרושים

פרק 4: יסודות התכנות ב'

- תכנות בעזרת משתנים: שם של משתנה, ערך של משתנה, המשתנה כקופסה; הוראת ההשמה; שימוש במשתנים לחישובים אריתמטיים פשוטים; העברת ערכים בין משתנים; מעקב אחר ביצוע תכניות בסיוע של טבלאות מעקב; הוראות קלט; סוגי משתנים; הכנת תרגיל מסכם
- אלגוריתמים ומבני בקרה בסיסיים: לולאות מונה המכילות הוראות השמה; מושג האלגוריתם כפתרון למשפחה של בעיות דומות; אלגוריתמים לחישוב סכום וממוצע; הוראת תנאי: אלגוריתמים למנייה מותנית ולמציאת מקסימום/מינימום

פרק 5: כלי תוכנה ויישומיהם

- הכרת הגיליון האלקטרוני – פרק חובה
- הכרת הגיליון ותכונותיו, שימוש בעזרה (Help) של התכנית, הכנסת מידע לגיליון ודפדוף בו
- יישומים חישוביים בגיליון: הכנסת נוסחאות פשוטות: הפקודות COPY ו-MOVE הפונקציות INT, MIN, MAX, AVG, SUM, COUNT; כתובות יחסיות ומוחלטות (ללא כתובות מעורבות); הוראת IF בסיסית; טעינה ושמירה)
- הכרת מסד המידע: הכרת מסד המידע ותכונותיו, הכנסת מידע למסד ודפדוף בו; עדכון מידע, הדפסת דוח, טעינה ושמירה
- הכרת התמלילן: הכרת התמלילן ותכונותיו, כתיבה ועריכה; הדפסה; טעינה ושמירה; הכרת הסביבה הגרפית; הכרת המחולל הגרפי ותכונותיו, ציור ועריכת סרטוט; הדפסה; טעינה ושמירה

נושאי הלימוד במדעי המחשב
2 יחידות לימוד (השלמה ל-3 יח"ל)
שאלון מספר 602

כללי

נבחנים המעוניינים להיבחן במדעי המחשב בהיקף של 3 יח"ל, צריכים להיבחן בשני שאלונים: בשאלון מספר 602 המפורט להלן, וכן בשאלון מספר 601.

כדי לקבל ציון סופי במקצוע **מדעי המחשב** בהיקף של 3 יח"ל, יש לקבל 55 נקודות לפחות בשאלון מספר 602.

בתכנית הלימודים של השאלון הזה שני נושאים:

נושא א' – יסודות מדעי המחשב 1

נושא ב' – יסודות מדעי המחשב 2

פרקי הלימוד בכל אחד מן הנושאים מפורטים להלן.

נושא א' – יסודות מדעי המחשב 1

מטרות היחידה

- להקנות מושגי יסוד ועקרונות שעליהם מושתת תחום מדעי המחשב
- ללמוד את המושגים, הבעיה האלגוריתמית והאלגוריתם
- לפתור בעיות באמצעות אלגוריתמים
- ליישם את המושג 'אלגוריתם' על-ידי כתיבת תכניות בשפת תכנות עילית והרצתן על מחשב
- לשמש יחידת לימוד העומדת בפני עצמה, אך גם בסיס להמשך לימוד מדעי המחשב בבית-הספר התיכון

פרקי הלימוד

פרק 1: מבוא

פרק 2: מודל חישוב בסיסי

פרק 3: מבוא לפיתוח אלגוריתמים

פרק 4: ביצוע מותנה

פרק 5: נכונות של אלגוריתמים

פרק 6: ביצוע חוזר

פרק 7: יעילות של אלגוריתמים

פרק 8: תת-משימות: פונקציות/פעולות

פרק 9 : מערכים חד-ממדיים
פרק 10 : שילוב והרכבה של מבני בקרה

סביבת העבודה

שפות התכנות האפשריות ביחידה זו הן ג'אווה ו-C#.

ניתן להשתמש בכל סביבת עבודה בהתאם לשפת התכנות. היחידה נכתבה בצורה גנרית ואינה מחייבת שימוש במהדר או בסביבה מסוימת.

פרק 1: מבוא

מטרות הפרק

- לחשוף את הנבחנים למושגי יסוד של עיבוד נתונים אוטומטי, דהיינו : שדה, רשומה, קובץ. הדבר יכול להיעשות תוך ניתוח בעיה אלגוריתמית או בנפרד
- לחשוף את הנבחנים למושגים המרכזיים של היחידה – 'אלגוריתם' ו'בעיה אלגוריתמית'
- להדגים אלגוריתמים (וכן לא-אלגוריתמים) באמצעות פסאודו-קוד או בכתיבה מילולית, כולל התניות וביצוע חוזר

פירוט התכנים

- אלגוריתם ; בעיה אלגוריתמית, תהליך הביצוע המושרה על-ידי אלגוריתם וקלטים. מחשב ; חומרה ; תוכנה ; מערכת הפעלה ; שפת תכנות ; הידור ; הרצה
- אלגוריתמים (וכן לא-אלגוריתמים) באמצעות פסאודו-קוד או בכתיבה מילולית, כולל התניות וביצוע חוזר (לא יילמדו הנושאים : הוראות בקרה, הוראות פעולה וכו')

פרק 2: מודל חישוב בסיסי

מטרות הפרק

להכיר את מודל החישוב של משתנים ושינוי ערכים ואת האלגוריתם הסדרתי הפשוט במודל זה

פירוט התכנים

- נתונים ; משתנים ; קלט ; פלט ; אלגוריתם פשוט כסדרה של הוראות לשינוי ערכים ; תהליך הביצוע ; מצבים בזמן ביצוע ; פיתוח אלגוריתם 'שטוח' (זיהוי קלטים ופלטים, זיהוי ותיעוד משתנים, בחירת טיפוסי משתנים)
- פונקציות/פעולות ספרייה בסיסיות : ערך מוחלט, ערך שלם (כל הפונקציות), ריבוע, שורש ריבועי, מספר אקראי. סעיף זה אינו עומד בפני עצמו אלא כמשולב תוכן

- היכרות עם סביבת העבודה של שפת התכנות. עבודה בסביבת העבודה: טיפוסי משתנים, הגדרת משתנים, הוראות השמה, ביטויים חשבוניים, הוראות קלטים ופלט, מבנה תכנית בסיסית
  הערה: הדוגמאות יכללו גם אלגוריתמים מילוליים וגם קטעי תכנית. ביצוע תכנית יודגם באמצעות טבלאות מעקב (מושג הנכונות יוזכר בהקשר זה, אך יילמד לפרטיו רק בפרק 5). אין להיכנס לתחביר מפורט של ביטויים. יש להדגיש שבחירת טיפוסי המשתנים היא חלק מפיתוח התכנית ותיעודה.

פרק 3: מבוא לפיתוח אלגוריתמים

מטרות הפרק

להמחיש את הצורך במבני בקרה ובאלגוריתמים מורכבים, ותוך כדי כך לתרגל פירוק בעיה לתת-בעיות פשוטות יותר

פירוט התכנים

פירוק בעיה לתת-בעיות, מנגנוני בנייה של אלגוריתמים מורכבים מאלגוריתמים פשוטים יותר
הערה: תוצגנה דוגמאות המראות שלא תמיד די באלגוריתם פשוט (סדרתי). יודגם הקשר הקיים בין התת-בעיות שבפירוק לבין חלקי האלגוריתם.

פרק 4: ביצוע מותנה

מטרות הפרק

- ללמוד את מבנה הבקרה של ביצוע מותנה בצורות שונות ואת מרכיביו
- להכיר אלגוריתמים שמשתמשים בהם בביצוע מותנה

פירוט התכנים

תנאי; ביצוע מותנה; קשר and וקשר or; טבלאות אמת ל-and ו-or; תנאי בוליאני עם יחסי סדר; משפט if (בגרסה מפוצלת ובגרסה עוקפת); תרגול בהבנת תכניות וכתיבתן
הערה: יושם דגש על המושגים 'התניה', 'פיצול' ו'עקיפה', ועל העובדה שלא כל קטעי האלגוריתם מבוצעים בכל הרצה. בדוגמאות ובתרגילים יופיעו תנאים המכילים and ו-or, אך לא ההרכבות שלהם. אין ללמד not, אך יש לתרגל שימוש ביחסים שיש להם שלילות מפורשות בשפה, כגון > ושלילתו =<.

פרק 5: נכונות של אלגוריתמים

מטרות הפרק

- להכיר בהכרה ראשונית את המושג נכונות שילווה את חומר הלימוד כולו
- להכיר את היתרונות ואת החסרונות של בדיקה באמצעות הרצה של אלגוריתם/תכנית
- להכיר את החשיבות של בניית אלגוריתם בצורה נאותה ושל תיעוד הולם (כולל תיעוד הקלטים החוקיים)
- להכיר את ההבדל בין שגיאה לוגית באלגוריתם לבין שגיאה הנובעת מטעות במימוש האלגוריתם בשפת התכנות

הערות

- נבחני הרמה הרגילה יתמקדו בבדיקת הפלט עבור דוגמאות שונות של קלט, הפקת פלט על-פי קלט נתון, אימות האלגוריתם על-ידי טבלת מעקב, אפיון קלטים לפלט נתון.
- נבחני הרמה המוגברת (5 יח"ל) ילמדו את כל הפרק כפי שהוא מופיע בתכנית הלימודים.
- בכיתות הטרוגניות יילמד הפרק על-פי הרמה הרגילה.

פירוט התכנים

- תחום הקלטים החוקיים; נכונות ביחס לבעיה אלגוריתמית וקלטיה; אלגוריתם נכון ואלגוריתם שגוי; בדיקת אלגוריתם באמצעות הרצה על קלטי בדיקה ומגבלותיה של בדיקה כזאת; תיעוד ככלי עזר להשגת נכונות ובדיקת נכונות
- תכנית שגויה (שגיאות תחביר, שגיאות בזמן ריצה, פלטים שגויים)
- הרצה ובדיקה של תכניות קיימות

פרק 6: ביצוע חוזר

מטרות הפרק

- ללמוד את מבנה הבקרה של ביצוע חוזר
- לתרגל ביצוע חוזר ככלי לעידון אלגוריתמים
- להבחין בין כתיבה אלגוריתמית של לולאה לבין מימושה בשפת תכנות
- להשתמש בלולאות לאלגוריתמים המצריכים מנייה או צבירה

פירוט התכנים

ביצוע חוזר: מקרים נפוצים (פעולה אחת החוזרת על עצמה, פעולה החוזרת ומתבצעת על נתונים שונים, ביצוע חוזר עם פעילות מצטברת); מונים; צוברים; תנאי סיום; ביצוע אינסופי; משפט while, תנאי בוליאני כזקיף

הערות

- הצד היישומי בפרק זה יתרכז במשפט ה-while. יש להדגים גם באמצעות פסאודו-קוד ניטרלי שאין לו תרגום ישיר לשפת התכנות, למשל: "בצע לכל איבר בקבוצה". במקרים כאלה ניתן לתרגל גם באמצעים לא-ממוחשבים. יש להדגיש שבביצוע חוזר טקסט האלגוריתם נשאר קבוע, אך הוא משרה תהליכי ביצוע באורכים שונים (כולל אורך 0) התלויים בקלט. יש לקשור זאת לביצוע שאינו מסתיים (דהיינו בעל אורך אינסופי).
- יש להדגיש את המשמעות המיוחדת של מושג הנכונות בהקשר של ביצוע חוזר (מילוי אחר התנאים הנדרשים בסיום הביצוע החוזר והבטחת סיום הביצוע). ניתן להציג לולאה פשוטה "חזור n פעמים" (ממומשת על-ידי צורה פשוטה של for) לפני לימוד יסודי של לולאת while (אין הכוונה ללימוד ממשי ומעמיק של לולאת for).

פרק 7: יעילות של אלגוריתמים

מטרות הפרק

- להכיר הכרה ראשונית את מושג היעילות, שילווה את חומר הלימוד כולו
- להזכיר לנבחני הרמה הרגילה את המושג בלבד ולהעביר את יתרת השעות לפרק 10 – שילוב והרכבה של מבני בקרה

הערות

- נבחני הרמה המוגברת ילמדו את הפרק כפי שהוא מופיע בתכנית הלימודים.
- בכיתות הטרוגניות יילמד הפרק על-פי הרמה הרגילה.

פירוט התכנים

- זיהוי וספירת הוראות חשובות; זמן ביצוע כפונקציה של גודל הקלט; השוואה בין זמני ריצה, המקרה הגרוע ביותר
- השפעת האלגוריתם על זמן הריצה והשפעת היישום כולו (שפת התכנות, המהדיר והמחשב) על זמן הריצה
- מדידה והשוואה בין זמני ביצוע של תכניות קיימות תוך הכנסת שינויים

הערה: יש להדגים מהן ההוראות החשובות, או הדומיננטיות, לסוגים שונים של בעיות. אין ללמד סימון O-גדול. דוגמה אפשרית במעבדה היא בדיקת ראשוניות של מספר n באמצעות בדיקת המספרים 2, ..., n ולאחר מכן בדיקת 2, ..., $\sqrt{n}$ (אפשר להראות גם ביצועים של אלגוריתם הסתברותי לבעיה זו, והשיפור שהוא נותן, אך ללא הסבר). יש להדגיש את ההבדל שבין תכנית בעלת טקסט קצר לבין תכנית יעילה.

פרק 8: תת-משימות: פונקציות/פעולות

מטרות הפרק

- ללמוד את מושג הפונקציה/הפעולה כאמצעי לפתרון בעיה בעזרת פתרון תת-בעיות
- ליישם פונקציות/פעולות בשפת התכנות

פירוט התכנים

- שימוש בפונקציה/בפעולה כהתמודדות עם תת-בעיה; אלגוריתם לפונקציה/לפעולה כפתרון התת-בעיה; קריאה לפונקציה/לפעולה כהוראה באלגוריתם הקורא
- פונקציות/פעולות בשפת התכנות: הגדרת פונקציה/פעולה, גוף הפונקציה/הפעולה, פרמטר ערך, משתנה מקומי

  הערה: יש לקשור את החומר עם המושגים העיוניים של פרק 3. יש לשלב פונקציות/פעולות הנכתבות על-ידי הנבחנים עם פונקציות/פעולות מערכת. יש להדגיש את הצורך בנכונות הפונקציה/הפעולה הנקראת ביחס להנחות הקלט שלה.

פרק 9: מערכים חד-ממדיים

מטרות הפרק

- להכיר את הצורך במבני נתונים
- ללמוד את מבנה הנתונים מערך כאוסף לינארי של משתנים מאותו סוג

פירוט התכנים

מבני נתונים והצורך בהם; מערך; מציין: הגדרת מערך; טיפוס המערך; מציינים; טווח המציינים; טיפוסי מציינים. משפט for; טיפוס תווי ומערך תווי

הערה: יש להדגיש את ההקבלה שבין משתנה לתא של מערך, ואת העובדה שמערך נבנה כדי לאסוף יחד קבוצה של משתנים למטרה משותפת ולאפשר טיפול אלגוריתמי אחיד (להדגים באמצעות השוואה בין טיפול בשלושה משתנים לבין טיפול באלף). יש להראות את הקשר ההדוק שבין ביצוע חוזר כמנגנון בנייה של אלגוריתם לבין מערך כמנגנון ארגון של נתונים. יש להדגיש את הקשר שבין משפט for למערך חד-ממדי ואת תפקידו הכפול של המציין בהקשר זה.

פרק 10: שילוב והרכבה של מבני בקרה

מטרות הפרק

- לתרגל את כל הכלים שנלמדו בפרקים הקודמים ולהעמיק בהם
- להכיר את בעיית החיפוש

פירוט התכנים

- פתרון בעיות מורכבות באמצעות שילוב והרכבה של מבני בקרה
- טיפוס בוליאני; קשר not; קינון ושילוב מנגנונים (while מקונן, if ו-while משתלבים, מנגנוני בקרה בתוך פונקציות וכדומה)

הערה: יש להדגים ולתרגל את הנלמד באמצעות בעיות מתקדמות ומורכבות יותר (לדוגמה: בעיות על מערכים ממוינים או כמעט ממוינים (לא מיון), חיפוש סדרתי, מציאת זוגות שאינם בסדר הנכון וכדומה). למתקדמים אפשר להוסיף חיפוש בינרי. ניתן להרחיב בנושא של תווים בעזרת דוגמאות מעולם עיבוד התמלילים. יש להעמיק ולהרחיב גם בנושאי יעילות ובסוגים שונים של ביצועים חוזרים ושילובם. יש להעמיק, באמצעות פסאודו-קוד, גם בניתוח ובפירוק של בעיה לפני היישום בשפת התכנות.

נושא ב' – יסודות מדעי המחשב 2

מטרות היחידה

- להעמיק בחומר שנלמד ב״יסודות מדעי המחשב 1' ולהרחיבו
- להוסיף כלים לפיתוח ולמימוש אלגוריתמים, למשל פרוצדורות/פעולות, רקורסיה ומערכים רב-ממדיים
- להעמיק את הדיון בנושאי נכונות ויעילות
- להציג אלגוריתמים לבעיות חשובות (למשל מיון), המשמשות אמצעי למידה והדגמה לנושאים שנלמדו

פרקי הלימוד

פרק 1: פיתוח אלגוריתמים

פרק 2: פעולות

פרק 3: תווים ומחרוזות

פרק 4: בעיות אלגוריתמיות מתקדמות

פרק 5: יעילות ונכונות של אלגוריתמים – הרחבה

פרק 6: טיפוסים ומבוא למבני-נתונים

פרק 7: המחלקה, התאמה ל-C# ולג'אווה

סביבת העבודה

שפות התכנות האפשריות ביחידה זו הן ג'אווה ו-C#.

ניתן להשתמש בכל סביבת עבודה בהתאם לשפת התכנות. היחידה נכתבה בצורה גנרית ללא תלות במהדר זה או אחר.

פרק 1: פיתוח אלגוריתמים

מטרות הפרק

להעמיק בניתוח של בעיה ופתרונה ולתרגלה

פירוט התכנים

ניתוח בעיה מורכבת יותר ופיתוח פתרונה בשלבים מלמעלה למטה (top-down); תכנות מודולרי

פרק 2: פעולות

פירוט התכנים

חזרה על פעולות בעלות ערך החזרה וערך החזרה ריק; חזרה על העברת פרמטרים; scope של משתנה; תיעוד של פעולה, כולל ציון טענות הכניסה ויציאה לפעולה

הערה: יש לקשור לכאן את הנלמד בפרק 8 של 'יסודות מדעי המחשב 1'.

פרק 3: תווים ומחרוזות

מטרות הפרק

להקנות כלים לפתרון בעיות לעיבוד טקסט

פירוט התכנים

- מחרוזות; יחס סדר מילוני; ייצוג מחרוזות כמערכי תווים; בניית אלגוריתמים עבור מחרוזות
- שימוש בפעולות פשוטות של המחלקה String: אורך מחרוזת, השוואת מחרוזות, העתקת מחרוזות, מחיקת מחרוזות, שרשור מחרוזות, מציאת תת-מחרוזת

פרק 4: בעיות אלגוריתמיות מתקדמות

מטרות הפרק

- להכיר את בעיית המיון והמיזוג ואלגוריתמים שונים לפתרונן
- לתרגל נושאים הנלמדים ביחידה ולהעמיק בהם

פירוט התכנים

חיפוש ומיון; מיזוג מערכים ממוינים; בעיות נוספות, כולל כאלה המשתמשות במספרים אקראיים

הערות

- יש לשלב כאן פרוצדורות/פונקציות/פעולות מערכת נוספות שטרם נלמדו ככלי עזר לפתרון הבעיות הנ"ל.
- יש ללמד שתי שיטות חיפוש: עם זקיף וחיפוש בינרי. נבחני הרמה המוגברת ילמדו לפחות שתי שיטות מיון – מיון-דחיפה ומיון-בועות. חיפוש בינרי ילמדו רק נבחנים ברמה המוגברת.

פרק 5: יעילות ונכונות של אלגוריתמים – הרחבה

פרק רשות לנבחני הרמה הרגילה (3 יח"ל); פרק חובה לנבחני הרמה המוגברת

מטרות הפרק

- להעמיק בלימוד נושא היעילות והנכונות של אלגוריתמים
- ללמוד כלים ושיטות לניפוי שגיאות

פירוט התכנים

ספירת מספר הביצועים של פעולות עיקריות בהתאם לסוג הבעיה (למשל: במיון – מספר ההשוואות; בבדיקת ראשוניות – מספר פעולות הכפל והחילוק); ניפוי של חלקי תכניות לפני הרכבתם לתכנית אחת; ניפוי חלקי תכניות על-ידי שימוש בפיגומים (תכנית עזר). שימוש בכלים ממוחשבים לניפוי שגיאות.

הערה: בעת בניית האלגוריתם לפתרון בעיה יש לקשר בין פירוק בעיה לתת-בעיות לבין בדיקת הנכונות של האלגוריתם על חלקיו.

פרק 6: טיפוסים ומבוא למבני-נתונים

מטרות הפרק

- להכיר את המושג 'טיפוס נתונים'
- להכיר את מבנה הנתונים 'רשומה'
- לבנות מבנים מורכבים

פירוט התכנים

- המושג טיפוס: הצהרת טיפוס, טיפוס מפורט
- מבני נתונים מורכבים: מערכים. רשומות ושילובם; שדה ברשומה; הגדרת רשומות; גישה לשדה בתוך רשומה; קליטה לתוך רשומה; הגדרת מערך דו-ממדי; מציינים למערך דו-ממדי

פרק 7: המחלקה, התאמה ל-C# ולג'אווה

פרק רשות לנבחני הרמה הרגילה.

מומלץ לנבחני הרמה המוגברת לבצע בחופשת הקיץ עבודה מסכמת בנושא מחלקות, שתכלול מבני נתונים מורכבים ותת-תכניות עם פרמטרים.

מטרות הפרק

- לערוך היכרות מעמיקה עם המחלקה בסביבת העבודה
- לחדד את רעיון המודולריות של מערכת תוכנה באמצעות בנייה של מחלקות ובאמצעות שימוש בהן

פירוט התכנים

מחלקה בעלת תכונות פרטיות ותכונות מורכבות

פעולות של מחלקה: פעולות בונות, קובעות ומאחזרות, פעולות המקבלות פרמטרים

בניית מחלקה בסביבת העבודה; בניית פרויקט בסביבת העבודה, צירוף מחלקה לפרויקט

הערות (לגבי כל השפות הנלמדות)

- במערך חד-ממדי כלולות גם הפעולות הבאות: השוואה בין מערכים ממוינים, מיזוג מערכים ממוינים, חיפוש סדרתי למציאה או להוספה של איבר, פיצול מערך ל-2 מערכים, בניית מערך חלקי, לדוגמה: כאשר נתון מערך ממוין עם חזרות – יש לבנות מערך ללא חזרות.
- במערך דו-ממדי כלולות גם הפעולות הבאות: חיפוש סדרתי לפי שורות ולפי עמודות, בדיקת מיון בשורות, בעמודות ובאלכסונים ראשיים, בדיקת קיום תנאי בשורות, בעמודות ובאלכסונים ראשיים, השוואת שורה או עמודה למערך חד-ממדי, הוספת שורה או עמודה למערך, מציינים למערך דו-ממדי.
- נבחני הרמה המוגברת ילמדו גם קבצים של רשומות תוך הדגשת נושאים אלה: גישה סדרתית; אורך לא חסום; קבצים כזיכרון עמיד; זמן גישה והשפעה על זמני ביצוע של תכנית.

נושאי הלימוד במדעי המחשב
2 יחידות לימוד (השלמה ל-5 יח״ל)
שאלון מספר 603

כללי

נבחנים המעוניינים להיבחן במדעי המחשב בהיקף של 5 יח״ל, צריכים להיבחן בשלושה שאלונים: בשאלון מספר 603, בשאלון מספר 602, וכן בשאלון מספר 601.
כדי לקבל ציון סופי במקצוע **מדעי המחשב** בהיקף של 5 יח״ל, יש לקבל 55 נקודות לפחות בכל אחד מן השאלונים שמספריהם 602 ו-603.

בשאלון הזה שתי יחידות לימוד:
היחידה הרביעית – עיצוב תוכנה (יחידת חובה)
היחידה החמישית – הנבחנים יבחרו **באחד** מבין פרקי הבחירה האלה:

- מודלים חישוביים
- תכנות מונחה עצמים
- מבוא לחקר ביצועים
- מערכות מחשב ואסמבלר

היחידה הרביעית – עיצוב תוכנה

דרישות קדם

ידיעה טובה ושליטה בנושאים האלה: ניתוח בעיה וניסוח פתרונה באופן אלגוריתמי; מערכים ורשומות/מחלקות; פונקציות ופרוצדורות/פעולות, כולל העברת פרמטרים, על-ידי הפניה או על-ידי ערך; רקורסיה; לוגריתמים

אוכלוסיית היעד

נבחני הרמה המוגברת במדעי המחשב שסיימו לפחות את ׳יסודות מדעי המחשב 1׳ ואת ׳יסודות מדעי המחשב 2׳

מטרות היחידה

- להקנות את עיקרי הגישה המערכתית
- לפתח את החשיבה המופשטת על-ידי היכולת להגדיר כלים מורכבים מאלו הבסיסיים הניתנים על-ידי שפת התכנות, במיוחד על-ידי הגדרת טיפוסי נתונים מופשטים
- להכיר טיפוסי נתונים מופשטים ידועים (כגון: רשימה, מחסנית, תור, עץ בינרי) ושימוש בהם לפתרון בעיות נתונות
- להגדיר טיפוסי נתונים מופשטים חדשים ומימושם
- להקנות יכולת לנתח את יעילותם של אלגוריתמים ואת התכניות המממשות אותם
- להכיר אלגוריתמים המאפשרים פעולות מתקדמות (חיפוש ומיון) על טיפוסי נתונים שונים; לעשות בחינה השוואתית של יעילות האלגוריתמים הנלמדים ביחידה
- להקנות יכולת לבחור טיפוסי נתונים המתאימים למימוש פתרון לבעיה, להגדיר את הטיפוסים, להעמידם לרשות המשתמש על-ידי כתיבת ממשקים מתאימים ומימושם בשפת התכנות הנלמדת

פרקי הלימוד

פרק 1: מבוא ורקורסיה

פרק 2: המחלקה – חזרה

פרק 3: טיפוסי נתונים מופשטים

פרק 4: מחסנית

פרק 5: יעילות

פרק 6: רשימה

פרק 7: עץ בינרי

סביבת העבודה

שפות התכנות האפשריות ביחידה זו הן ג'אווה ו-C#.

ניתן להשתמש בכל סביבת עבודה בהתאם לשפת התכנות. היחידה נכתבה בצורה גנרית ללא תלות במהדר זה או אחר.

פרק 1: מבוא ורקורסיה

מטרות הפרק

- להכיר הכרה ראשונית את המושג רקורסיה ככלי לפתרון בעיות
- להכיר את היתרונות ואת החסרונות של כתיבה רקורסיבית
- להציג את הרקע הכללי ליחידת הלימוד 'עיצוב תוכנה'
- להציג את הצורך ואת התועלת שבחלוקת מערכת לתת-משימות
- להעמיק בשיטת התכנון מלמעלה למטה
- להכין את הנבחנים למשימה הנלווית על-ידי חשיפת הנבחנים למושגים הבסיסיים הדרושים לעיסוק במשימה

פירוט התכנים

- הנדסת תוכנה; תכנון מהכלל אל הפרט; מפרט מערכת; מודול; ממשק; מימוש; הסתרת מידע; שימוש חוזר בקוד; ממשק למשתמש
- מערכת תוכנה: תכונות (נכונות, עמידות, יעילות, תיעוד), תחזוקה, שדרוג
- רקורסיה: קריאה רקורסיבית; בסיס הרקורסיה; תנאי עצירה; מעקב על אלגוריתמים רקורסיביים; כתיבת אלגוריתם רקורסיבי
- להדגיש את הקשר בין הגדרות רקורסיביות (למשל, הגדרת עצרת או מספרי פיבונצ׳י), לבין התכונות הרקורסיבית שלהם; לציין יתרונות וחסרונות של תכונות רקורסיביות, מבחינת תהליך התכנות, זמן ריצה ומקום בזיכרון

  הערה: נבחני הרמה הרגילה ילמדו רק רקורסיה פשוטה, כגון עצרת, ולא רקורסיה כפולה כגון סדרת פיבונצ׳י. הנבחנים הנבחנים ביחידה הרביעית ילמדו גם רקורסיה כפולה (מתנדנדת) כגון סדרת פיבונצ׳י.

פרק 2: המחלקה – חזרה

פירוט התכנים

תרגול השימוש במחלקה בסביבת העבודה

פרק 3: טיפוסי נתונים מופשטים

מטרות הפרק

- לערוך היכרות עם המושג 'טיפוס נתונים מופשט'
- ללמוד ולתרגל שלבי הגדרה, ייצוג המימוש והשימוש בטיפוס נתונים מופשט
- להכיר את החשיבות של התייחסות למקרים חריגים ומקרי גבול ודרכי הטיפול בהם

פירוט התכנים

הצגת טיפוסי נתונים מוגדרים מראש, כגון מספר שלם; הגדרת טיפוס נתונים מופשט המכיל ערכים, פעולות ותחום הקיום שלהם; עבודה עם טיפוס נתונים דרך ממשק בסביבת העבודה; ייצוג של טיפוס נתונים וטיפול במגבלות הייצוג; מימוש של טיפוס נתונים מופשט בהתאם לשפת התכנות

פרק 4: מחסנית

מטרות הפרק

- לתרגל הגדרה של טיפוס נתונים מופשט
- לערוך היכרות עם טיפוס הנתונים המופשט 'מחסנית' על-ידי שימושים שונים
- לתרגל את מימוש טיפוס הנתונים 'מחסנית'

פירוט התכנים

הגדרת טיפוס הנתונים 'מחסנית', ערכים ופעולות; הכרת הממשק של מחסנית (ראו נספח); מימוש המחסנית על-ידי מערך

פרק 5: יעילות

מטרות הפרק

- להבין את המושג 'יעילות', באמצעות הכרת המדד של סיבוכיות זמן הריצה, ואת החשיבות של מדד זה
- ללמוד מושגים בסיסיים בתחום היעילות (המקרה הגרוע ביותר)
- לאבחן שמדד טוב לסיבוכיות הזמן של אלגוריתם הוא מספר הצעדים הבסיסיים שהאלגוריתם מבצע כתלות באורך הקלט
- להכיר את המושג 'סדר גודל' (O-גדול)
- לבצע ניתוח של סיבוכיות זמן ריצה של אלגוריתם

פירוט התכנים

- כיצד מודדים יעילות של אלגוריתם, מקום וזמן; ניתוח סיבוכיות זמן ריצה של אלגוריתמים; אורך קלט; צעד בסיסי; שיפור יעילותו של אלגוריתם בקבוע; סדר גודל; הכרת משפחות של סדרי גודל: לוגריתמי, לינארי, ריבועי ומעריכי; השוואת סדרי גודל שונים עבור אורכי קלט שונים; מקרה טוב, גרוע וממוצע; שיפור יעילות של אלגוריתם בסדר גודל; הבנת ההבדל בין שיפור בקבוע לעומת שיפור בסדר גודל; חיפוש סדרתי וחיפוש בינרי
- ניתוח יעילות של מיון-בועות ומיון-מיזוג (שנלמדו ב'יסודות מדעי המחשב 2')

  הערה: בכל הפרקים הבאים יש להתייחס ליעילותם של האלגוריתמים השונים בהתאם לדרכי הייצוג השונות.

פרק 6: רשימה

מטרות הפרק

- להכיר את טיפוס הנתונים 'רשימה'
- להפנים את העיקרון של הסתרת מידע באמצעות היכרות עם מימושים שונים של אותו ממשק
- להכיר את המנגנון של הקצאת זיכרון דינמית
- לממש טיפוס נתונים מופשט באמצעות שימוש בטיפוס נתונים קיים (מחסנית ותור המָמומשים בעזרת רשימה)

פירוט התכנים

- כתיבת ממשק לטיפוס הנתונים רשימה (ראו נספח); הגדרת המושג 'מקום ברשימה'; ייצוג רשימה על-ידי מערך; הקצאה זיכרון דינמית; ייצוג רשימה על-ידי שרשרת חוליות; מחסנית ותור כמקרים פרטיים של רשימה; מיון הכנסה
- השוואת יעילותם של אלגוריתמים שונים לפי דרכי הייצוג השונות

פרק 7: עץ בינרי

מטרות הפרק

- להכיר את טיפוס הנתונים 'עץ בינרי' ושימושים שונים שלו
- לתרגל את השימוש בשגרות רקורסיביות ולהעריך את יעילותן
- להכיר את עץ החיפוש הבינרי ואת השימוש בו למיון

פירוט התכנים

- הכרת טיפוס הנתונים 'עץ בינרי': אב, אב-קדמון, אח, בן (שמאלי, ימני), גובה, מסלול, עלה, עץ, עץ בינרי מלא, צאצא, צומת, קשת, רמה, שורש, תת-עץ (שמאלי, ימני); הכרת הממשק של טיפוס הנתונים 'עץ בינרי' (ראו נספח)
- סריקה בסדר סופי; סריקה בסדר תוכי; סריקה בסדר תחילי; סריקה לפי רמות; עץ חיפוש; מיון על-ידי עץ חיפוש

נספח – ממשקים בשפת ג'אווה

ממשק החוליה הגנרית <Node<T

המחלקה מגדירה חוליה גנרית שבה יש ערך מטיפוס T והפניה לחוליה העוקבת.

הפעולה בונה חוליה. הערך של החוליה הוא x, ואין לה חוליה עוקבת.	Node (T x)
הפעולה בונה חוליה. הערך של החוליה הוא x, והחוליה העוקבת לה היא next. ערכו של next יכול להיות null.	Node (T x, Node<T> next)
הפעולה מחזירה את הערך של החוליה.	T getInfo()
הפעולה מחזירה את החוליה העוקבת. אם אין חוליה עוקבת, הפעולה מחזירה null.	Node<T> getNext()
הפעולה משנה את הערך השמור בחוליה ל-x.	void setInfo (T x)
הפעולה משנה את החוליה העוקבת ל-next. ערכו של next יכול להיות null.	void setNext (Node<T> next)
הפעולה מחזירה מחרוזת המתארת את החוליה.	String toString()

יעילות הפעולות נקבעת על-פי המחלקה Node המופיעה בפרק 7 – ייצוג אוספים (ראו סעיף ב.5.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1).

ממשק המחלקה 'מחסנית' Stack<T>

המחלקה מגדירה טיפוס אוסף בעל פרוטוקול LIFO להכנסה ולהוצאה של ערכים.

Stack()	הפעולה בונה מחסנית ריקה.
boolean isEmpty()	הפעולה מחזירה 'אמת' אם המחסנית הנוכחית ריקה, ו'שקר' – אחרת.
void push (T x)	הפעולה מכניסה את הערך x לראש המחסנית הנוכחית (דחיפה).
T pop()	הפעולה מוציאה את הערך שבראש המחסנית הנוכחית ומחזירה אותו (שליפה). **הנחה**: המחסנית הנוכחית אינה ריקה.
T top()	הפעולה מחזירה את הערך שבראש המחסנית הנוכחית בלי להוציאו. **הנחה**: המחסנית הנוכחית אינה ריקה.
String toString()	הפעולה מחזירה תיאור של המחסנית, כסדרה של ערכים, במבנה הזה (x_1 הוא האיבר שבראש המחסנית): $[x_1, x_2, \ldots, x_n]$.

יעילות הפעולות נקבעת על-פי המחלקה Stack המופיעה בפרק 8 – מחסנית ותור. המחלקה מיוצגת בעזרת שרשרת חוליות (ראו סעיף ה.3.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה toString() המתבצעת בסדר גודל לינארי.

ממשק המחלקה 'תור' Queue<T>

המחלקה מגדירה טיפוס אוסף בעל פרוטוקול FIFO להכנסה ולהוצאה של ערכים.

Queue()	הפעולה בונה תור ריק.
boolean isEmpty()	הפעולה מחזירה 'אמת' אם התור הנוכחי ריק, ו'שקר' – אחרת.
void insert (Tx)	הפעולה מכניסה את הערך x לסוף התור הנוכחי.
T remove()	הפעולה מוציאה את הערך שבראש התור הנוכחי ומחזירה אותו. **הנחה**: התור הנוכחי אינו ריק.
T head()	הפעולה מחזירה את ערכו של האיבר שבראש התור בלי להוציאו. **הנחה**: התור הנוכחי אינו ריק.
String toString()	הפעולה מחזירה מחרוזת המתארת את התור כסדרה של ערכים, במבנה הזה (x_1 הוא האיבר שבראש התור): $[x_1, x_2, \ldots, x_n]$.

יעילות הפעולות נקבעת על-פי המחלקה Queue המופיעה בפרק 8 – מחסנית ותור. המחלקה מיוצגת בעזרת שרשרת חוליות והפניה לזנב התור (ראו סעיף ט.2.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה toString() המתבצעת בסדר גודל לינארי.

ממשק המחלקה 'רשימה' List<T>

המחלקה מגדירה אוסף סדרתי-לינארי, שהגישה אל ערכיו מתבצעת בכל מקום באוסף.

הפעולה בונה רשימה ריקה.	List()
הפעולה מחזירה 'אמת' אם הרשימה הנוכחית ריקה, ו'שקר' – אחרת.	boolean is Empty()
הפעולה מחזירה את המקום של החוליה הראשונה ברשימה הנוכחית. אם הרשימה ריקה, הפעולה מחזירה null.	Node<T> getFirst()
הפעולה מכניסה לרשימה הנוכחית את הערך x, מקום אחד אחרי המקום pos. אם pos הוא null, x יוכנס למקום הראשון ברשימה. הפעולה מחזירה את המקום של החוליה החדשה שהוכנסה. הנחה : pos הוא מקום ברשימה הנוכחית או null.	Node<T> insert (Node<T> pos, T x)
הפעולה מוציאה מן הרשימה הנוכחית את האיבר הנמצא במקום pos, ומחזירה את המקום העוקב ל-pos. אם הוצא האיבר האחרון – יוחזר null. הנחה : pos הוא מקום ברשימה הנוכחית, ואינו null.	Node<T> remove (Node<T> pos)
הפעולה מחזירה תיאור של הרשימה, כסדרה של ערכים, במבנה הזה (x1 הוא האיבר הראשון ברשימה) : $[x_1, x_2, \ldots, x_n]$.	String toString()

יעילות הפעולות נקבעת על-פי המחלקה List המופיעה בפרק 9 – רשימה. המחלקה מיוצגת בעזרת שרשרת חוליות (ראו סעיף ד).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולות toString() ו- remove(...) המתבצעות בסדר גודל O(n).

ממשק המחלקה 'חוליה בינרית' BinTreeNode<T>

המחלקה מגדירה חוליה בינרית שבה יש ערך מטיפוס T ושתי הפניות לחוליות בינריות.

הפעולה בונה חוליה בינרית. ערך החוליה הוא x וערך שתי ההפניות שלה הוא null.	BinTreeNode (T x)
הפעולה בונה חוליה בינרית שערכה יהיה x. left ו-right הן (הפניות אל) הילד השמאלי והילד הימני שלה. ערכי ההפניות יכולים להיות null.	BinTreeNode (BinTreeNode<T> left, T x, BinTreeNode<T> right)
הפעולה מחזירה את הערך של החוליה.	T getInfo()
הפעולה משנה את הערך השמור בחוליה ל-x.	void setInfo (T x)
הפעולה מחזירה את הילד השמאלי של החוליה. אם אין ילד שמאלי, הפעולה מחזירה null.	BinTreeNode<T> getLeft()
הפעולה מחזירה את הילד הימני של החוליה. אם אין ילד ימני, הפעולה מחזירה null.	BinTreeNode<T> getRight()
הפעולה מחליפה את הילד השמאלי בחוליה left.	void setLeft (BinTreeNode<T> left)
הפעולה מחליפה את הילד הימני בחוליה right.	void setRight (BinTreeNode<T> right)
הפעולה מחזירה מחרוזת המתארת את הערך השמור בחוליה.	String toString()

יעילות הפעולות נקבעת על-פי המחלקה BinTreeNode המופיעה בפרק 10 – עץ בינרי (ראו סעיף ג.2.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1).

ממשק המחלקה מפה Map<V>

המחלקה מגדירה אוסף דינמי הממפה מפתחות וערכים. המפתחות במפה יהיו מטיפוס מחרוזת והערכים יהיו גנריים.

הפעולה בונה מפה ריקה.	Map()
הפעולה מחזירה את הערך הקשור למפתח key. הפעולה מחזירה null אם המפתח אינו קיים במפה.	V getValue (String key)
הפעולה מוסיפה למפה הנוכחית את המפתח key ואת הערך value הקשור אליו. אם המפתח key קיים במפה, הפעולה מעדכנת את הערך הקשור אליו בערך value שהתקבל.	void insert (String key, V value)
הפעולה מוציאה מן המפה הנוכחית את המפתח key ואת הערך הקשור אליו. הפעולה מחזירה את	V remove (String key)

הערך הקשור למפתח שהוצא מן המפה. אם המפתח אינו קיים במפה היא מחזירה null.	
הפעולה מחזירה את אוסף המפתחות שקיימים במפה הנוכחית ממוין בסדר אלפביתי עולה. אם המפה ריקה, יוחזר מערך בגודל אפס.	String[] getAllKeys()
הפעולה מחזירה מחרוזת המתארת את המפה כך: [key1:value1, key2:value2, key3:value3,…]	String toString()

יעילות הפעולות נקבעת על-פי המחלקה Map המופיעה בפרק 11 – מפה. המחלקה מיוצגת בעזרת עץ-חיפוש-בינרי (ראו סעיף ד.3.).

O(1)	Map()
O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.	V getValue (String key)
O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.	void insert (String key, V value)
O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.	V remove (String key)
O(n)	String[] getAllKeys()
O(n)	String toString()

נספח – ממשקים – C#

ממשק החוליה הגנרית Node<T>

המחלקה מגדירה חוליה גנרית שבה יש ערך מטיפוס T והפניה לחוליה העוקבת.

Node (T x)	הפעולה בונה חוליה. הערך של החוליה הוא x, ואין לה חוליה עוקבת.
Node (T x, Node<T> next)	הפעולה בונה חוליה. הערך של החוליה הוא x והחוליה העוקבת לה היא next. ערכו של next יכול להיות null.
T GetInfo()	הפעולה מחזירה את הערך של החוליה.
Node<T> GetNext()	הפעולה מחזירה את החוליה העוקבת. אם אין חוליה עוקבת, הפעולה מחזירה null.
void SetInfo (T x)	הפעולה משנה את הערך השמור בחוליה ל-x.
void SetNext (Node<T> next)	הפעולה משנה את החוליה העוקבת ל-next. ערכו של next יכול להיות null.
string ToString()	הפעולה מחזירה מחרוזת המתארת את החוליה.

יעילות הפעולות נקבעת על-פי המחלקה Node המופיעה בפרק 7 – ייצוג אוספים (ראו סעיף ב.5.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1).

ממשק המחלקה 'מחסנית' Stack<T>

המחלקה מגדירה טיפוס אוסף בעל פרוטוקול LIFO להכנסה ולהוצאה של ערכים.

Stack()	הפעולה בונה מחסנית ריקה.
bool IsEmpty()	הפעולה מחזירה 'אמת' אם המחסנית הנוכחית ריקה, ו'שקר' – אחרת.
void Push (T x)	הפעולה מכניסה את הערך x לראש המחסנית הנוכחית (דחיפה).
T Pop()	הפעולה מוציאה את הערך שבראש המחסנית הנוכחית ומחזירה אותו (שליפה). הנחה : המחסנית הנוכחית אינה ריקה.
T Top()	הפעולה מחזירה את הערך שבראש המחסנית הנוכחית בלי להוציאו. הנחה : המחסנית הנוכחית אינה ריקה.
string ToString()	הפעולה מחזירה תיאור של המחסנית, כסדרה של

ערכים, במבנה הזה (x1 הוא האיבר שבראש המחסנית) : $[x_1, x_2, \ldots, x_n]$.	

יעילות הפעולות נקבעת על-פי המחלקה Stack המופיעה בפרק 8 – מחסנית ותור. המחלקה מיוצגת בעזרת שרשרת חוליות (ראו סעיף ה.3.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה ToString() המתבצעת בסדר גודל לינארי.

ממשק המחלקה 'תור' Queue<T>

המחלקה מגדירה טיפוס אוסף בעל פרוטוקול FIFO להכנסה ולהוצאה של ערכים.

הפעולה בונה תור ריק.	Queue()
הפעולה מחזירה 'אמת' אם התור הנוכחי ריק, ו'שקר' – אחרת.	bool IsEmpty()
הפעולה מכניסה את הערך x לסוף התור הנוכחי.	void Insert (Tx)
הפעולה מוציאה את הערך שבראש התור הנוכחי ומחזירה אותו. הנחה : התור הנוכחי אינו ריק.	T Remove()
הפעולה מחזירה את ערכו של האיבר שבראש התור בלי להוציאו. הנחה : התור הנוכחי אינו ריק.	T Head()
הפעולה מחזירה מחרוזת המתארת את התור כסדרה של ערכים, במבנה הזה (x1 הוא האיבר שבראש התור) : $[x_1, x_2, \ldots, x_n]$.	string ToString()

יעילות הפעולות נקבעת על-פי המחלקה Queue המופיעה בפרק 8 – מחסנית ותור. המחלקה מיוצגת בעזרת שרשרת חוליות והפניה לזנב התור (ראו סעיף ט.2.).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולה ToString() המתבצעת בסדר גודל לינארי.

ממשק המחלקה 'רשימה' <List<T>

המחלקה מגדירה אוסף סדרתי-לינארי, שהגישה אל ערכיו מתבצעת בכל מקום באוסף.

הפעולה בונה רשימה ריקה.	List()
הפעולה מחזירה 'אמת' אם הרשימה הנוכחית ריקה, ו'שקר' – אחרת.	bool IsEmpty()
הפעולה מחזירה את המקום של החוליה הראשונה ברשימה הנוכחית. אם הרשימה ריקה, הפעולה מחזירה null.	Node<T> GetFirst()
הפעולה מכניסה לרשימה הנוכחית את הערך x, מקום אחד אחרי המקום pos. אם pos הוא null, x יוכנס למקום הראשון ברשימה. הפעולה מחזירה את המקום של החוליה החדשה שהוכנסה. הנחה: pos הוא מקום ברשימה הנוכחית או null.	Node<T> Insert (Node<T> pos, T x)
הפעולה מוציאה מן הרשימה הנוכחית את האיבר הנמצא במקום pos, ומחזירה את המקום העוקב ל-pos. אם הוצא האיבר האחרון – יוחזר null. הנחה: pos הוא מקום ברשימה הנוכחית ואינו null.	Node<T> Remove (Node<T> pos)
הפעולה מחזירה תיאור של הרשימה, כסדרה של ערכים, במבנה הזה (x1 הוא האיבר הראשון ברשימה): $[x_1, x_2, \ldots, x_n]$.	string ToString()

יעילות הפעולות נקבעת על-פי המחלקה List המופיעה בפרק 9 – רשימה. המחלקה מיוצגת בעזרת שרשרת חוליות (ראו סעיף ד).

כל הפעולות מתבצעות בסדר גודל קבוע, O(1), למעט הפעולות ()ToString ו-(...)Remove המתבצעות בסדר גודל O(n).

ממשק המחלקה 'חוליה בינרית' BinTreeNode<T>

המחלקה מגדירה חוליה בינרית שבה יש ערך מטיפוס T ושתי הפניות לחוליות בינריות.

הפעולה בונה חוליה בינרית. ערך החוליה הוא x וערך שתי ההפניות שלה הוא null.	BinTreeNode (T x)
הפעולה בונה חוליה בינרית שערכה יהיה x. left ו-right הן (הפניות אל) הילד השמאלי והילד הימני שלה. ערכי ההפניות יכולים להיות null.	BinTreeNode (BinTreeNode<T> left, T x, BinTreeNode<T> right)
הפעולה מחזירה את הערך של החוליה.	T GetInfo()
הפעולה משנה את הערך השמור בחוליה ל-x.	void SetInfo (Tx)
הפעולה מחזירה את הילד השמאלי של החוליה. אם אין ילד שמאלי, הפעולה מחזירה null.	BinTreeNode<T> GetLeft()
הפעולה מחזירה את הילד הימני של החוליה. אם אין ילד ימני, הפעולה מחזירה null.	BinTreeNode<T> GetRight()
הפעולה מחליפה את הילד השמאלי בחוליה left.	void SetLeft (BinTreeNode<T> left)
הפעולה מחליפה את הילד הימני בחוליה right.	void SetRight (BinTreeNode<T> right)
הפעולה מחזירה מחרוזת המתארת את הערך השמור בחוליה.	string ToString()

יעילות הפעולות נקבעת על-פי המחלקה BinTreeNode המופיעה בפרק 10 – עץ בינרי (ראו סעיף ג.2.).
כל הפעולות מתבצעות בסדר גודל קבוע, O(1).

ממשק המחלקה מפה Map<V>

המחלקה מגדירה אוסף דינמי הממפה מפתחות וערכים. המפתחות במפה יהיו מטיפוס מחרוזת והערכים יהיו גנריים.

הפעולה בונה מפה ריקה.	Map()
הפעולה מחזירה 'אמת' אם המפתח key קיים במפה הנוכחית, ו'שקר' – אחרת.	bool ContainsKey (string key)
הפעולה מחזירה את הערך הקשור למפתח key. הנחה: המפתח קיים במפה.	V GetValue (string key)
הפעולה מוסיפה למפה הנוכחית את המפתח key ואת הערך value הקשור אליו. אם המפתח key קיים במפה, הפעולה מעדכנת את הערך הקשור אליו בערך value שהתקבל.	void Insert (string key, V value)

V Remove (string key)	הפעולה מוציאה מן המפה הנוכחית את המפתח key ואת הערך הקשור אליו. הפעולה מחזירה את הערך הקשור למפתח שהוצא מן המפה. הנחה: המפתח קיים במפה.
string[] GetAllKeys()	הפעולה מחזירה את אוסף המפתחות שקיימים במפה הנוכחית ממוין בסדר אלפביתי עולה. אם המפה ריקה, יוחזר מערך בגודל אפס.
string ToString()	הפעולה מחזירה מחרוזת המתארת את המפה כך: [key1:value1, key2:value2, key3:value3,…]

יעילות הפעולות נקבעת על-פי המחלקה Map המופיעה בפרק 11 – מפה. המחלקה מיוצגת בעזרת עץ-חיפוש-בינרי (ראו סעיף ד.3.).

Map()	O(1)
bool ContainsKey (string key)	O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.
V GetValue (string key)	O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.
void Insert (string key, V value)	O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.
V Remove (string key)	O(log n) – במקרה הממוצע, O(n) – במקרה הגרוע.
string[] GetAllKeys()	O(n)
string ToString()	O(n)

היחידה החמישית – פרקי בחירה

מודלים חישוביים

אוכלוסיית היעד

נבחנים אשר למדו מדעי המחשב ברמה רגילה

מטרות היחידה

לערוך היכרות עם תחום תאורטי של מדעי המחשב, המתאר מכונות חישוב באמצעות כמה מודלים ומנתח את כוחם ואת תכונותיהם של מודלים אלה

פרקי הלימוד

פרק 1: תיאור מערכות ופתרון חידות

פרק 2: אוטומט סופי דטרמיניסטי

פרק 3: מילים ושפות פורמליות

פרק 4: מודלים נוספים של אוטומט סופי

פרק 5: אוטומט המחסנית

פרק 6: כוחו ומגבלותיו של מודל אוטומט המחסנית

פרק 7: מכונת טיורינג

ביבליוגרפיה

1. האוניברסיטה הפתוחה (1991), **אוטומטים ושפות פורמליות**, כרכים א' ו-ב'.
2. הראל ד' (1991), אלגוריתמיקה, **יסודות מדעי המחשב**, האוניברסיטה הפתוחה.
3. Barwise, Etchemendy (1993), **Turing's World, An Introduction to Computability Theory**, CLSI Publications.
4. Davis, Sigal, Weyuker (1994), **Computability, Complexity and Languages, Fundamentals of Theoretical Computer Science**, Academic Press.
5. Hopcroft, Ullman (1979), **Introduction to Automata Theory, Languages and Computations**, Addison-Wesel.

יחידה זו מחולקת לשלושה חלקים:

חלק א': האוטומט הסופי

חלק ב': אוטומט המחסנית

חלק ג': מכונת טיורינג

חלק א': האוטומט הסופי (פרקים 1–4)

לחלק הזה מוקדש רוב יחידת הלימוד הזאת. בחלק זה מוקנים לנבחנים הכלים, דרכי החשיבה והמונחים המקובלים בתחום, תוך כדי עיסוק במשפחת השפות הרגולריות (באמצעות האוטומטים הסופיים). המודלים שמוצגים בחלק זה הם האוטומט הסופי הדטרמיניסטי, האוטומט הסופי הדטרמיניסטי הלא-מלא והאוטומט הסופי הלא-דטרמיניסטי.

חלק ב': אוטומט המחסנית (פרקים 5–6)

חלק זה עוסק במשפחת השפות חופשיות ההקשר ומציג אותה באמצעות מודל אוטומט המחסנית.

חלק ג': מכונת טיורינג (פרק 7)

חלק זה מציג מודל לתכנית מחשב – מכונת טיורינג – ונערך בו דיון על התזה של צ'רץ' וטיורינג ועל מגבלותיו של המחשב.

פרק 1: תיאור מערכות ופתרון חידות

מטרות הפרק

להכיר את המושגים הבסיסיים בתחום

פירוט התכנים

תיאור גרפי של מערכות: דוגמאות ומושגים (מצב, קלט, מעבר, מצב התחלתי); פתרון חידות בעזרת תיאור גרפי: דוגמאות ומושגים (מצב מקבל, מצב מלכודת)

פרק 2: אוטומט סופי דטרמיניסטי

מטרות הפרק

- להציג את מודל האוטומט הסופי הדטרמיניסטי
- לתרגל בניית אוטומטים

פירוט התכנים

אוטומט סופי דטרמיניסטי; מסלול חישוב מקבל ולא מקבל; תיאור אוטומט בדרך גרפית או על-ידי פירוט מרכיביו תוך כדי שימוש בטבלת מעברים או בפונקציית מעברים; אוטומטי ספירה, חיפוש

פרק 3: מילים ושפות פורמליות

מטרות הפרק

1. להכיר מושגים בסיסיים בתורת השפות הפורמליות
2. לחקור את כוחו של מודל האוטומט הסופי הדטרמיניסטי ואת התכונות של משפחת השפות הרגולריות

פירוט התכנים

מושגים בסיסיים: אות, א״ב, מילה, אורך מילה, המילה הריקה, שפה פורמלית; פעולות על מילים ועל שפות: שרשור, חזקה, היפוך; שפה רגולרית, שפות שאינן רגולריות, תכונות סגירות של משפחת השפות הרגולריות: דיון בסגירות לחלקיות, משלים, חיתוך ואיחוד

פרק 4: מודלים נוספים של אוטומט סופי

מטרות הפרק

- להבין כיצד אפשר – על-ידי שינוי הגדרה קיימת של מודל חישובי – לקבל מודלים חדשים
- להכיר את מושג האי-דטרמיניזם ולדון בהשוואת כוחם של מודלים חישוביים

פירוט התכנים

אוטומט סופי דטרמיניסטי לא-מלא; אוטומט סופי לא-דטרמיניסטי; שקילות של מודל האוטומט הסופי הדטרמיניסטי ושל מודל האוטומט הסופי הלא-דטרמיניסטי; תכונות סגירות של משפחת השפות הרגולריות: דיון בסגירות לשרשור, היפוך ואיחוד

פרק 5: אוטומט המחסנית

מטרות הפרק

- להכיר את מודל אוטומט המחסנית הלא-דטרמיניסטי
- לתרגל את בניית אוטומט המחסנית

פירוט התכנים

שימוש במחסנית כמבנה עזר, אוטומט מחסנית לא-דטרמיניסטי

פרק 6: כוחו ומגבלותיו של מודל אוטומט המחסנית

מטרות הפרק

1. הכרת כוחו ומגבלותיו של מודל אוטומט המחסנית
2. להשוות בין המודל החדש למודל האוטומט הסופי

פירוט התכנים

אוטומט מחסנית דטרמיניסטי; השוואה בין כוח החישוב של אוטומט מחסנית לא-דטרמיניסטי לבין אוטומט מחסנית דטרמיניסטי, משפחת השפות חופשיות ההקשר; שפות שאינן חופשיות הקשר; תכונות סגירות של משפחת השפות חופשיות ההקשר: דיון בסגירות חלקיות, משלים, חיתוך, איחוד, שרשור, היפוך

פרק 7: מכונת טיורינג

מטרות הפרק

- להציג מכונת טיורינג כמודל לתכנית מחשב
- להכיר את התזה של צ'רץ' וטיורינג

פירוט התכנים

מכונת טיורינג: הגדרה, דוגמאות ותרגילים, אי-עצירה של מכונות טיורינג, מכונות טיורינג שמחשבות פונקציות, השקילות של תכנית מחשב ומכונת טיורינג, התזה של צ'רץ' וטיורינג, בעיית העצירה

תכנות מונחה עצמים

סביבת ג׳אווה #C ,

היחידה פותחה על-ידי צוות מדעי המחשב – המרכז להוראת המדעים. אנו מודים לפרופ׳ כתריאל בארי ולגב׳ עופרה ברנדס שעמדו בראש הצוות שפיתח את היחידה.

מטרות היחידה

- לחשוף את הנבחנים לגישה המודרנית בתחום עיצוב תוכנה ותכנות – ׳תכנות מונחה עצמים׳
- להקנות לנבחנים ידע מקיף בתכנות על-פי הגישה החדשה: הכרת העקרונות והמנגנונים העיקריים הכוללים מחלקות, העמסה, הגדרה מחדש, המרות
- לפתח את יכולת החשיבה המופשטת בעזרת הרעיונות המתקדמים של הגישה: ירושה, פולימורפיזם וממשקים

הערה: שפת היישום המשמשת את היחידה היא ג׳אווה. אולם כיוון שרוב ההבדלים בין ג׳אווה ל- #C הם תחביריים, ומכיוון שבמגמת הנדסת תוכנה אפשר ללמוד Web Services בכל שפות ה- NET כל האמור לגבי ג׳אווה נאמר גם עבור #C.

דרישות קדם

היחידה מהווה קורס המשך לתכנית יסודות 1 ו-2 וליחידה ׳עיצוב תוכנה׳. היא מסתמכת על התכנים הנלמדים ביחידות אלה וממשיכה לעסוק בהם.

הקורס מניח כי הנבחנים מבינים היטב את נושאי היסוד: כתיבה אלגוריתמית, פיתוח פתרונות מובנים, הכרת מבני בקרה ושליטה, לולאות ופתרון בעיות מורכבות כמיון וחיפוש, הכרת הנושא העוסק בטיפוסי נתונים מופשטים והכרת המושגים האלה: הסתרת מידע, הכמסה, עבודה עם ממשקים ושימוש חוזר בקוד.

אוכלוסיית היעד

נבחנים הלומדים מדעי המחשב בהיקף מוגבר (לפחות 5 יח״ל)

סביבת העבודה בג׳אווה

סביבת העבודה שנבחרה לצורך לימוד היחידה היא JCreator. הסביבה היא חלונאית ומיישמת רבים מן התפקודים המוכרים לנבחנים מעבודתם במחשב. הסביבה משמשת כעורך ומכילה מהדר (קומפיילר) ואת כל ספריית קובצי ה-API הקיימת בג׳אווה. בסביבת העבודה ניתן גם להריץ את התכניות. סביבת העבודה – וכן כל הקבצים והתוכנות אשר נלווים אליה ונדרשים לצורך עבודה תקינה – מותקנים אוטומטית על-ידי תקליטור הנמסר לכל נבחן.

סביבת העבודה ב-#C

סביבת העבודה היא סביבת NET. בכל פעם שתכנית הלימודים מתייחסת לתחביר ג'אווה או לסביבת העבודה JCreator, יש להמיר לפקודה המתאימה (אם יש צורך) ב-#C או לסביבת NET.

ביבליוגרפיה

הספרות העוסקת בתכנות מונחה עצמים רבה ורחבה, הן במישור התאורטי שלה והן בלימוד שפת ג'אווה. כמו-כן, רבים האתרים המציגים את התחום ומציעים תרגול והתנסות בתחום זה. אנו להלן רק אחדים משלל האתרים והמקורות המוקדשים לנושא זה.

- **תכנות מונחה עצמים בשפת ג'אווה למתכנתי שפת C**, בית-הספר לטכנולוגיה של האוניברסיטה הפתוחה, מטח ומה"ט. (ספר קריא מאוד שאינו מצריך ידע מוקדם ב-C)
- Bradly, L. Johes. (2003), **#C**, סדנת לימוד, הוצאת הוד עמי (עברית, 725 עמודים)
- אלבהארי בן, דרייטון פיטר, בראד מריל (2003), **יסודות שפת #C**, הוצאת אופוס (עברית, 202 עמודים)
- אתר האינטרנט של חברת SUN אשר מפתחת את ג'אווה. זהו אתר מומלץ מאוד המציע תרגול וסקירת נושאים, ומתעדכן באופן שוטף. כתובתו היא: http://java.sun.com/docs/books/tutorial/java/
- אתר האינטרנט של חברת Microsoft אשר מפתחת את #C. זהו אתר מומלץ מאוד המציע תרגול וסקירת נושאים, ומתעדכן באופן שוטף. כתובתו היא: http://www.msdn.microsoft.com/vcsharp/
- הקורס 'מבוא לתכנות מונחה עצמים בג'אווה' – המופיע באתרי האינטרנט של מרבית האוניברסיטאות והמכללות הפועלות בישראל
-

Reges, Stuart (2003), **Can C# Replace Java in CS1 and CS2?,** University of Arizona, Computer Science Department site, http://www.cs.arizona.edu/~reges/sigcse/csharp.pdf

(מאמר העוסק בהבדלים שבין ג'אווה ל- #C)

פרקי הלימוד

פרק 1:	כל העולם כולו עצמים
פרק 2:	עוברים לג'אווה
פרק 3:	על המחלקה, העצמים ומה שביניהם
פרק 4:	פענוח צפונות ה-()main
פרק 5:	ירושה ופולימורפיזם
פרק 6:	ממשקים
פרק 7:	שפות תכנות: משפות מכונה עד ג'אווה
פרק הרחבה:	מחלקות מופשטות

פרק 1: כל העולם כולו עצמים

מטרות הפרק

- לשמש פרק מבוא
- להציג את מושגי היסוד ואת העקרונות של גישת 'התכנות מונחה העצמים'
- לשים דגש על מגוון רחב של דוגמאות המקשרות את הגישה לחיי היום-יום של הנבחנים, ולידע קודם מלימוד מדעי המחשב

פירוט התכנים

- 'תכנות מונחה עצמים' – Object Oriented Programming, צורת החשיבה
- מהו עצם – object, ולמה הוא משמש
- איברי עצם – תכונות (attributes) – כמתארות מצב, ושיטות (methods) – כמתארות יכולת פעולה
- תהליך הפיתוח של תכנית מונחית עצמים (הדגמה וסקירה)
- תקשורת בין עצמים: פנייה לתכונות ולשיטות
- מחלקה – class: כתבנית מופשטת המגדירה טיפוס; המחלקה כמשמשת ליצירת מופעים (instances); השיטה הבונה (constructor)
- ירושה – דרך ליצירת יחסי סיווג; עיקרון בתכנות מונחה עצמים; פולימורפיזם – בקצרה
- רעיונות כלליים המוכרים לנבחן ועומדים בבסיס הגישה: הסתרת מידע, הכמסה, עבודה עם ממשקים, שימוש חוזר בקוד

פרק 2: עוברים לג'אווה

מטרות הפרק

- להקנות לנבחנים את היכולת לתכנת בשפת ג'אווה ברמה מקבילה כמעט ליכולת התכנות שהייתה להם בשפה הפרוצדורלית
- ללמד את כללי הכתיבה (המוסכמות) בשפת ג'אווה
- ללמד מהם מבני בקרה ולהקנות לנבחנים שליטה בזרימת התכנית של שפת ג'אווה
- להכיר את סביבת העבודה –JCreator , ולהתנסות בה

פירוט התכנים

- דקדוק, תחביר ומוסכמות: נקודה-פסיק, סימון בלוק פקודות, הערות, הזחה (Indent), מילים שמורות ראשונות; מוסכמות כתיבה ראשונות בג'אווה (שם מחלקה, שמות משמעותיים)
- שיטות: סימון-הנקודה (dot notation); פקודות הדפסה פשוטות:
  System.out.print(),System.out.println()
- טיפוסי נתונים בסיסיים: int, double, char, boolean
- הצהרת משתנים והצבת ערכים במשתנים

- המרה בין טיפוסים בסיסיים (casting)
- קבועים – final ומוסכמות כתיבה
- שליטה בזרימת התכנית: תנאי (if-else), לולאת for, לולאת while, לולאת do-while, פקודת switch. סימן השוויון == לעומת סימן ההצבה =
- סביבת העבודה JCreator: התקנה, יצירת פרויקט: פתיחה וסגירה, כתיבת תכנית, הידור והרצה, העתקות וצירופים של קבצים וספריות; תרגול

פרק 3: על המחלקה, העצמים ומה שביניהם

מטרות הפרק

- לחזור על המושגים היסודיים שנלמדו בפרק המבוא, תוך כדי מימושם בשפת ג'אווה
- לעשות שימוש ראשוני במושגי המחלקה, העצם והתקשורת בין עצמים
- לכתוב תכניות ראשונות ומחלקות ראשונות

פירוט התכנים

- המחלקה: הגדרה ויצירה. כותרת, מוסכמות בכתיבת שמות, הרשאות גישה (באופן כללי)
- תכונות: אפיון, הגדרה ואתחול
- שיטות: חתימה, העברת פרמטרים, ערכי החזרה
- יצירת עצמים ממחלקה – השיטה הבונה (constructor), קונסטרקטור בררת המחדל, שימוש בפקודה new
- הפניות – משתנים המכילים עצמים על-ידי הפניות (references), העברת עצמים כפרמטרים
- שימוש בעצמים קיימים, תקשורת בין עצמים, שימוש בסימון-הנקודה
- עצמים מורכבים
- העמסת שיטות (overloading), שיטה-בונה-מעתיקה (copy constructor)
- עבודה עם ממשקים. הכרה ותרגול של שימוש בג'אווה API

פרק 4: פענוח צפונות ה-main()

מטרות הפרק

- לסקור נושאים תחביריים מתקדמים (מערכים, מחרוזות ואיברי מחלקה)
- להעמיק בנושא הרשאות גישה
- להכיר את מנגנון התיעוד של ג'אווה (Javadoc)

פירוט התכנים

- מערכים חד-ממדיים, המערך כעצם בעל תכונות ושיטות, יצירה ואתחול – ייחודו של העצם מערך, חריגה מגבולות מערך, מערך של עצמים; מערכים דו-ממדיים ומערכים לא-ריבועיים (בקצרה)

- המחלקה String – מחרוזת כעצם; שיטות וממשק המחלקה, אופרטור השרשור; השיטה toString() כדרך להדפסת אובייקטים
- יבוא מחלקות: השימוש בפקודה – import
- איברי מחלקה; תכונות ושיטות סטטיות (static); הגדרה ופנייה אל איברי מחלקה; מחלקות שירות
- הרשאות גישה (access specifiers): פומבי – public ופרטי – private; מימוש הרעיונות הסתרת המידע והכימוס; הפרדה בין ממשק למימוש
- מנגנון ה-Javadoc, כמאפשר את רעיון העבודה עם ממשקים והסתרת המידע; שימוש

פרק 5: ירושה ופולימורפיזם

מטרות הפרק

להכיר שני עקרונות יסוד בתכנות מונחה עצמים: ירושה – inheritance, ופולימורפיזם – polymorphism

פירוט התכנים

- ירושה – מייצגת חלוקה היררכית טבעית-אנושית: תהליך של מיון וסיווג היררכי; ייצוג היחס 'סוג של' (is a)
- ירושה בג'אווה (single inheritance) – התוספת extends בכותרת המחלקה, מחלקת-על (super class) ותת-מחלקה (sub class), הרשאת הגישה 'מוגן' – protected
- ירושה – מה עובר? שיטות בונות, השימוש ב- super לגבי שיטה רגילה ולגבי קונסטרקטור
- הגדרה מחדש של שיטות – overriding
- ירושה מן המחלקה הראשונה – Object
- ירושה כתומכת ברעיון הסתרת המידע, שימוש חוזר בקוד, עבודה עם ממשקים
- מעט על תכנון מונחה עצמים תוך שימוש ברעיון הירושה
- פולימורפיזם – רב-צורניות: היתרון, העוצמה של הרעיון והשימוש בו
- זימון פולימורפי של שיטות, המרה למעלה (upcasting), המרה למטה (downcasting)
- האופרטור instanceof הקיים בג'אווה, תוך כדי הדגשה של צמצום השימוש בו לטובת רעיון הפולימורפיזם
- מחלקות עוטפות – הדרך להפוך טיפוסים בסיסיים לאובייקטים

פרק 6: ממשקים

מטרות הפרק

- להכיר את מנגנון הממשקים (interfaces) בג'אווה
- להבין כיצד המנגנון מאפשר שימוש מתקדם בעקרון הפולימורפיזם

פירוט התכנים

- ממשק כמגדיר התנהגות ומייצג את היחס 'מתפקד כ-'. (לעומת 'סוג של' בירושה)
- ממשק אינו מחלקה: הוא מגדיר טיפוס אך אינו משמש כתבנית ליצירת עצמים
- הממשק כמגדיר חוזה; כל מחלקה המעוניינת לממש את הממשק – חייבת לממש את כל המפורט בחוזה; החובות והזכויות של המחלקות המממשות
- הממשק בג'אווה: תוספת ה- implements לכותרת מחלקה המממשת את הממשק
- ירושה בין ממשקים; מימוש מרובה של ממשקים
- ממשקים בשירות הפולימורפיזם

פרק 7: שפות תכנות: משפות מכונה עד ג'אווה

מטרות הפרק

- להציג את הרקע ההיסטורי והמחקרי להתפתחות של גישת התכנות מונחית העצמים
- להבין את הסדר הכרונולוגי של פיתוח שפות תכנות
- להבין את הרציונל ואת המטרות שעמדו בפני מפתחי שפות התכנות

הערה: זהו פרק העשרה בלבד

פירוט התכנים

- סקירה היסטורית של פיתוח שפות תכנות: שפות מכונה, סף, שפות עיליות שונות
- גישה תכנותית חדשה: תכנות מונחה עצמים; מערכת היא אוסף של מודולים המתקשרים ביניהם
- ג'אווה כשפה: ההתפתחות; המאפיינים: חסינות, אי-תלות בפלטפורמה שעליה רצה התכנית, ניידות
- ספריית המחלקות הסטנדרטית – ג'אווה API
- יישומים (Applications) ויישומונים (Applets)

פרק הרחבה: מחלקות מופשטות

מטרות הפרק

- להכיר את המנגנון של מחלקות מופשטות (abstract classes) בג'אווה
- להבין כיצד המנגנון מאפשר שימוש מתקדם והרחבה של עקרון הפולימורפיזם

פירוט התכנים

- הצורך בהגדרת מחלקות מופשטות: המחלקה המופשטת כמגדירה רעיון שאינו ניתן למימוש וליצירה של עצמים בשלב נתון
- המחלקה המופשטת בג'אווה: כותרת המחלקה; מימושים חלקיים; הגדרת שיטות מופשטות; שיטות בונות

- המחלקה המופשטת כמגדירה חוזה; כל מחלקה המעוניינת לממש את המחלקה – חייבת לממש את כל המפורט בחוזה; החובות והזכויות של המחלקות המממשות; הגדרה מחדש במחלקות היורשות
- מחלקות מופשטות בשירות הפולימורפיזם

מבוא לחקר ביצועים

אוכלוסיית היעד

נבחנים הלומדים לפחות 3 יח"ל במתמטיקה, ושסיימו 3 יח"ל במדעי המחשב

דרישות קדם

- לשלוט בניתוח ובפתרון של מערכת משוואות לינאריות – בדרך אלגברית ובדרך גרפית
- לנתח סיבוכיות זמן ריצה של אלגוריתם (פרק 5 ביחידה 'עיצוב תוכנה')

מטרות היחידה

- להכיר את גישת חקר הביצועים לפתרון בעיות אופטימיזציה בתחומים יישומיים שונים
- להבין את המושגים הבסיסיים, העקרונות והשיטות של הענף המרכזי של חקר ביצועים –תכנון לינארי (ללא חובת שימוש באלגברה לינארית שאינו מקצוע קדם)
- להבין את המושגים הבסיסיים, את העקרונות ואת השיטות של בעיות זרימה ברשתות – המהוות נושא מרכזי בתחומים הבאים: אלגוריתמיקה, תורת הגרפים והרשתות, אופטימיזציה קומבינטורית, חקר ביצועים
- להיעזר בעקרונות של התכנון הלינארי להבנה מעמיקה יותר של התכונות של בעיות זרימה ברשתות, והשיטות לפתרונן; להיעזר בעקרונות של עיצוב תוכנה למימוש, לניתוח ולשיפור של שיטות אלה
- להתנסות בניסוח מודל מתמטי מתאים ובפתרון של בעיות אופטימיזציה בתחומים יישומיים שונים, כולל ניתוח היעילות של הפתרון

פרקי הלימוד

פרק 1: מודל התכנון הלינארי

פרק 2: פתרון של בעיות תכנון לינארי

פרק 3: בעיית התובלה

פרק 4: מודלים של זרימה ברשתות

פרק 5: בעיית המסלול הקצר ביותר

פרק 6: עץ פורש מינימלי

ביבליוגרפיה

מבוא לאלגוריתמים, כרך ב', האוניברסיטה הפתוחה 1997.

מודלים דטרמיניסטיים בחקר ביצועים, כרכים א' ו-ב', האוניברסיטה הפתוחה 1995.

J. Gal-Ezer, G. Zwas, "An Algorithmic Approach to Linear Systems", Int. J. Math. Educ. **Sci. Technol**. 1984 (pp. 501–519).

H. A. Taha, **Operations Research: An Introduction**, 6th edition, Prentice Hall 1996.

R.K. Ahuja, T.L. Magnanti and J.B. Orlin, **Network Flows: Theory, Algorithms and Applications**, Prentice-Hall 1993.

פתרונות לתרגילים המופיעים בספר האחרון נמצאים באתר האינטרנט הזה:

http://web.mit.edu/jorlin/www/SolutionManual/SolutionManual.html

אתרים ברשת

אנימציה של אלגוריתמים:

http://www-b2.is.tokushima-u.ac.jp/~ikeda/suuri/simplex/Simplex.html

http://www.cs.oswego.edu/~birgit/html/diplom/links.html

http://carnap.ss.uci.edu/java/dijkstra/DijkstraApplet.html

מילון מונחים בתורת הגרפים:

http://www.utm.edu/departments/math/graph/glossary.html#degree

אתרי קורסים המכילים חומר לימודי (class notes):

http://ubmail.ubalt.edu/~harsham/opre640/opre640.htm

http://www-math.mit.edu/~vempala/18.433/course99.html

http://wwwpub.utdallas.edu/~chandra/documents/7313.htm

http://www.mat.uc.pt/~eqvm/links/cursos.html

אתרים כלליים ופורטלים לחקר ביצועים:

http://www.opsresearch.com

http://www.opsresearch.com/OR-Links/index.html

http://mat.gsia.cmu.edu

http://www.math.rpi.edu/~mitchj/sites_or.html

פרק 1: מודל התכנון הלינארי

מטרות הפרק

- להכיר את ההנחות ואת המושגים הבסיסיים של מודל התכנון הלינארי
- לתרגל ניסוח בעיות בעזרת מודל התכנון הלינארי

פירוט התכנים

- דוגמאות של בעיות תכנון לינארי: הקצאת משאבים אופטימלית (רווח מקסימלי ברמת משאבים נתונה), מינימום עלות ברמת שירות נדרשת, ועוד
- ניסוח מתמטי של בעיית תכנון לינארי
- המרכיבים העיקריים של בעיית תכנון לינארי: משתני החלטה (רציפים), פרמטרים, אילוצים, אילוצי אי-שליליות, פונקציית מטרה
- ההנחות שעליהן מבוסס מודל התכנון הלינארי
- דוגמאות לניסוח בעיות תכנון לינארי

פרק 2: פתרון של בעיות תכנון לינארי

מטרות הפרק

- להציג ולתרגל את הפתרון הגרפי של בעיית תכנון לינארי בעלת שני משתנים
- לחקור את התכונות העיקריות של בעיות תכנון לינארי בעלות שני משתנים
- להציג את ההשלכות של הפתרון הגרפי על התכונות של בעיות תכנון לינארי בעלות N משתנים
- להכיר את העקרונות הבסיסיים של שיטת הסימפלקס
- להדגים את השלבים השונים בשיטת הסימפלקס

פירוט התכנים

- פתרון גרפי של בעיית תכנון לינארי בעלת שני משתנים; תיאור גרפי של התחום האפשרי במקרים הבאים: תחום חסום, תחום לא-חסום, תחום ריק; התכונות של התחום האפשרי: קמירות, נקודות קיצון; תיאור גרפי של פונקציית המטרה; היטלי גובה, התכונות של הפתרון האופטימלי, והמצבים האפשריים: פתרון יחיד, פתרונות מרובים, פתרון לא-חסום
- שיטת הסימפלקס: האלגברה של שיטת הסימפלקס והשלבים בשיטת הסימפלקס, התכונות ה'גאומטריות' של התחום האפשרי והתכונות של הפתרון האופטימלי, קריטריון האופטימליות: פתרונות בסיסיים אפשריים סמוכים; המצבים האפשריים: פתרון יחיד, פתרונות מרובים, פתרון לא-חסום
- סיכום שלבי הפתרון.

פרק 3: בעיית התובלה

מטרות הפרק

- להציג את המודל של בעיית התובלה ולהכיר את המושגים הבסיסיים שלה
- להציג כיצד ניתן לנצל את התכונות של בעיית התובלה למימוש יעיל של שיטת הסימפלקס ולפתרונה
- לתרגל את המימוש של שיטת הסימפלקס באמצעות פתרון בעיות תובלה; להציג בצורה לא פורמלית מושגים של בעיות זרימה ברשתות (מבוא לפרקים הבאים)

פירוט התכנים

- ניסוח בעיית התובלה כבעיית תכנון לינארי בשלמים
- התכונות של בעיית התובלה: תכונת הפתרונות השלמים, תכונת הפתרון האפשרי
- ניצול התכונות של בעיית התובלה למימוש יעיל של שיטת הסימפלקס: מציאת פתרון בסיסי אפשרי בשיטת פינה צפון מערבית, בדיקת אופטימליות, שיפור הפתרון
- סיכום היתרונות של השימוש בשיטת הסימפלקס המותאמת לבעיית התובלה לעומת השימוש בשיטת הסימפלקס לצורך בעיית תכנון לינארי כללית
- רשות: סקירת שיטות לשיפור היעילות של פתרון בעיית התובלה

פרק 4: מודלים של זרימה ברשתות

מטרות הפרק

- להדגים מודלים של בעיות זרימה ברשתות ולהכיר את המושגים הבסיסיים של בעיות אלו
- להציג את הגרף כטיפוס נתונים מופשט
- להדגים אלגוריתמים של בעיות זרימה ברשתות

פירוט התכנים

- דוגמאות של בעיות זרימה ברשתות
- הגדרת המרכיבים העיקריים של בעיות זרימה ברשתות: גרף-קדקוד, קשת, מסלול, עץ, מעגל
- רשת – זרימה, מחירים, קיבולת, היצע וביקוש
- סקירת מבני נתונים שונים לייצוג גרפים ורשתות
- מטריצת מסלולים

פרק 5: בעיית המסלול הקצר ביותר

מטרות הפרק

- להציג את בעיות המסלול הקצר ביותר ולהכיר את המושגים הבסיסיים של בעיות אלו
- להציג ולנתח יעילות של שיטות שונות לפתרון בעיות המסלול הקצר ביותר
- לתרגל ניסוח ופתרון של בעיות המסלול הקצר ביותר

פירוט התכנים

- דוגמאות של בעיות המסלול הקצר ביותר בתחומים שונים; סיווג של בעיות המסלול הקצר ביותר
- הגדרה פורמלית של הבעיה (כולל ההנחות) וניסוח כבעיית תכנון לינארי בשלמים, תכונת הפתרונות השלמים
- מציאת המסלולים הקצרים מקדקוד נתון לכל הקדקודים האחרים – אלגוריתם דיקסטרה; ניתוח היעילות של האלגוריתם
- הצגה יעילה של המסלולים הקצרים מקדקוד נתון לכל הקדקודים האחרים – עץ המסלולים הקצרים
- מציאת המסלולים הקצרים מקדקוד נתון לכל הקדקודים האחרים, כאשר לכל הקשתות משקל זהה – סריקת גרף על-ידי חיפוש לרוחב – BFS; עץ המסלולים הקצרים, עץ פורש BFS; ניתוח היעילות של האלגוריתם
- סריקת גרף לעומק – DFS; עץ פורש DFS, ניתוח היעילות של האלגוריתם, אלגוריתם למציאת רכיבים קשירים בגרף לא-מכוון ואלגוריתם למציאת רכיבי קשירות חזקה (רק״חים) בגרפים מכוונים
- מיון טופולוגי
- מציאת המסלולים הקצרים ביותר בין כל זוגות הקדקודים; אלגוריתמים של דיקסטרה, פלויד-וורשל

פרק 6: עץ פורש מינימלי

מטרות הפרק

- להכיר את המושגים הבסיסיים של בעיית העץ הפורש
- להציג ולנתח את היעילות של שיטות שונות לפתרון בעיית העץ הפורש

פירוט התכנים

- הצגת הבעיה
- עצים, הגדרות ותכונות יסוד
- האלגוריתם של קרוסקאל
- האלגוריתם של פרים

מערכות מחשב ואסמבלר

אוכלוסיית היעד

נבחנים שסיימו לפחות את "יסודות מדעי המחשב 1" ואת "יסודות מדעי המחשב 2"

מבוא ורציונל

יחידה זו מציגה את הקשר בין תוכנה לחומרה, ואת העקרונות של פיתוח תכניות בשפת סף.

מטרות היחידה

- להכיר את העקרונות של המבנה הבסיסי של מחשב, המשמש כמכונת חישוב אוניברסלית, עקרונות אשר דרושים ליישום התפיסה של תכנית מאוחסנת
- להבין את התפקידים השונים של מרכיבי המחשב בהקשר של אחסון תכנית וביצועה
- לדעת כיצד מאוחסן מידע (נתונים והוראות) במחשב
- להכיר את מבנה המחשב המבוסס על מעבד 8086
- להכיר את שפת הסף 8086 – את היתרונות של כתיבה בשפת סף לעומת כתיבת שפה עילית, את היתרונות ואת החסרונות הנובעים מן ההבדלים בין השקיפות של חומרת המחשב בשפות עיליות לעומת שפת סף
- להכיר את אוצר הפקודות בשפת הסף 8086, המדגימים מרכיבים שונים שקיימים בכל שפת תוכנה: הוראות השמה, בקרה (הסתעפות), לולאות, שגרות, קלט ופלט
- להבין כיצד מאוחסנים ומטופלים במחשב טיפוסי נתונים שונים (שלם, ממשי, תו) ומבני נתונים (מערכים)
- להבין כיצד מתבצעת תכנית בשפת סף ובשפות עיליות במחשב
- להכיר התפתחויות שחלו במחשבים מודרניים ושנועדו לשכלל את מבנה המחשב הבסיסי
- להכיר סוגיות בארכיטקטורה של מיקרומעבדים מודרניים (פנטיום), אשר נועדו להשיג מהירויות עיבוד גבוהות

סביבת העבודה

שפת התכנות ביחידה זו היא שפת טורבו אסמבלר. סביבת עבודה זו כוללת את debbugger ו- editor ומאפשרת להמחיש את העקרונות העיוניים והמעשיים של היחידה.

ביבליוגרפיה

1. **ארגון המחשב ושפת סף**, האוניברסיטה הפתוחה.
2. פיקז, אריה, **ארגון המחשב ותכנותו**, האוניברסיטה הפתוחה.
3. **שפת סף 8086/88**, בית-הספר לטכנולוגיה של האוניברסיטה הפתוחה.
4. **מחשבים ומיקרומעבדים**, חלקים א' ו-ב', בית-הספר לטכנולוגיה של האוניברסיטה הפתוחה, מטח ומשרד החינוך.

5. Uffenbeck, J., 1998, **The 80x86 Family, Design, Progrmming, and Interfacing**, Prentice-Hall International, Inc.

פרקי הלימוד

פרק 1 : מבוא

פרק 2 : הצגת המידע במחשב ושיטות ספירה

פרק 3 : המבנה הבסיסי של המחשב

פרק 4 : מבוא לשפת סף

פרק 5 : תכנות מתקדם בשפת סף של המיקרומעבד 8086

פרק 6 : ההתפתחות של מחשבים מודרניים

יחידה זו מחולקת לשלושה חלקים :

חלק א׳: החומרה (פרקים 1–3)

חלק זה מתאר את הארכיטקטורה של המחשב, המבוססת על תפיסה של תכנית מאוחסנת המממשת מכונת חישוב אוניברסלית (מודל von Neumann). ארכיטקטורה זו מאפיינת הן מחשבים המשמשים למטרות כלליות וקיימים זה עשרות שנים, והן מחשבים מודרניים ביותר. מימוש תפיסה זו מאפשר למחשב להריץ מגוון של תכניות (אפליקציות), מאפשר לו לקלוט מגוון של נתונים (מספרים, מילים) ממקורות קלט שונים, ומאפשר לו להציג מגוון של קלטים (טקסט, גרפיקה, קול וכו׳). ידע זה יהווה בסיס לתכנות בשפת סף וימחיש את עקרונות התכנות שהנבחנים למדו בקורס יסודות. הצגת החומרה תיעשה בשלבים : בשלב הראשון נתייחס לתרשים המלבנים ולעקרונות של אופן הביצוע של תכנית מאוחסנת במחשב ; בשלב השני נחשוף את המבנה העקרוני של כל יחידה באמצעות סכמת מלבנים, באופן שיאפשר להציג מידע מדויק יותר על אופן הביצוע של התכנית ; לבסוף, נפרט כיצד התכנית מתבצעת במחשב (על-ידי תיאור של מחזורי הכתיבה והקריאה). בסיום חלק זה, נציג את המבנה של מחשב המבוסס על מיקרו-מחשב 8086, ונתאר את המרכיבים העיקריים במחשב אשר דרושים ללימוד שפת סף.

חלק ב׳: התוכנה (פרקים 4–5)

בחלק זה הנבחנים יכירו וילמדו לכתוב תכניות בשפת סף, שפה שבה ההוראות מתייחסות באופן ישיר יותר למבנה המחשב. היכרות עם שפת סף חשובה מכמה היבטים : 1. היא מאפשרת הבנה טובה יותר של מבנה המחשב. 2. היא מזמנת לנבחן היכרות עם שפת תכנות המתאימה לביצוע יישומים שבהם הזמן הוא קריטי (יישומי זמן אמת). 3. היא מספקת הבנה טובה יותר של מושגים שנרכשו ביסודות מדעי המחשב 1 ו-2, ואשר מתייחסים למבנים בסיסיים של שפת תכנות כגון הוראות השמה, הוראות בקרה, שגרות וכו׳. לשם כך הנבחנים יכירו ויתרגלו את שפת הסף של מעבד 8086, כדוגמה מייצגת של שפות סף.

חלק ג׳: ההתפתחות והארכיטקטורה של מיקרומעבדים מודרניים (פרק 6)

חלק זה יכלול הרחבות המתייחסות למערכות מחשב מודרניות, ויידונו בו ההתפתחויות בתחום המחשבים וסוגיות בארכיטקטורה שמטרתן להשיג מהירויות עיבוד גבוהות יותר, למשל : גודל מילה, אוגרים מיוחדים,

אוגרים לטיפול בנקודה צפה, מעבדים המשמשים לעיבודים מתמטיים, ארכיטקטורה של פנטיום, זיכרון מטמון, צינור הוראות.

פרק 1: מבוא

מטרות הפרק

להכיר את המבנה הבסיסי של המחשב, אשר מממש את התפיסה של תכנית מאוחסנת של von Neuman.

הערה: לימוד המחשב באופן זה מתקשר לתפיסת המחשב כמכונת חישוב אוניברסלית, תפיסה שההנבחנים מכירים הן בתור משתמשים באפליקציות רבות והן בתור מתכנתים בשפות עיליות.

פירוט התכנים

- מהו מחשב – מחשב כמכונת חישוב אוניברסלית, תפיסה של תכנית מאוחסנת (נתונים והוראות)
- המבנה הסכמתי של המחשב הבסיסי – תרשים מלבנים המתאר את היחידות השונות של המחשב ואת התפקידים שלהן, ללא פירוט של מבנה היחידות הללו
- העברת מידע במערכת – סוג המידע שזורם מיחידה ליחידה
- איך תכנית מתבצעת באופן עקרוני – מחזור ביצוע ההוראה fetch and execute
- היסטוריה – התפתחות (דורות) של מחשבים

פרק 2: הצגת המידע במחשב ושיטות ספירה

מטרות הפרק

- ללמוד על האופן שבו מאוחסן במחשב מידע מסוגים שונים (הוראות תכנית, נתונים, בקרה וכו') והסיבות לכך שהמידע מאוחסן במחשב בשיטת ספירה אשר שונה משיטת הספירה העשרונית
- ללמוד את העקרונות האלה:
  - הצגת מידע מסוגים שונים באופן אחיד, כך שיהיה אפשר לממש את עקרון המכונה האוניברסלית, גם אם בשפה עילית אנו מתייחסים לנתונים המוצגים בצורה מגוונת (מספרים, מחרוזות וכו')
  - הסיבות לשימוש בשיטת הייצוג הבינרי במחשב

פירוט התכנים

- סוגי מידע שמאוחסנים וזורמים במחשב – הוראות (של תכנית), נתונים, הוראות בקרה, כתובות
- שיטות ספירה (ייצוג) – עשרונית, בינרית והקסהדצימלית; סדרי גודל של מספרים בינריים – kB , MB , GB
- המרה משיטת הספירה העשרונית לשיטת הספירה הבינרית וההקסהדצימלית
- ייצוג מספרים שלמים עם סימן
- ייצוג מספרים ממשיים עם סימן
- ייצוג תווים

- פעולות חישוב בשיטות הספירה הבינרית וההקסהדצימלית: חיבור, חיסור וכפל
- פעולות לוגיות: And, Or, Not, Xor
- קידוד במחשב: ASCII, BCD, קוד משלים ל-2, קוד EBCDIC

פרק 3: המבנה הבסיסי של המחשב

מטרות הפרק

- להציג את היחידות הבסיסיות של המחשב ואת המבנה של כל אחת מהן כדי להבין כיצד תכנית מתבצעת בצורה מפורטת.
- להציג את המרכיבים הבסיסיים ואת התפקידים של כל יחידה, אשר מאפשרים לבצע תכנית במחשב
- להכיר מושגים חשובים וללמוד מהי שפת סף ומהו המבנה של מיקרו-מחשב 8086
- להתייחס לרכיבים כאל קופסאות שחורות (בסכמת מלבנים מפורטת) ולתפקיד שלהם במסגרת המערכת.

הערה: פרק זה כולל חמישה סעיפים: יע"מ, זיכרון, יחידות קלט ופלט, פסים להעברת מידע בין יחידות המחשב השונות, וסיכום המתאר את אופן ביצוע ההוראות וזרימת המידע בין היחידות השונות. הסעיף האחרון מציג את הארכיטקטורה של מחשב המבוסס על מיקרומעבד 8086 כדוגמה למימוש העקרונות הנלמדים.

פירוט התכנים

- יחידת העיבוד המרכזית – היע"מ – תפקידיו, מבנהו העקרוני, מרכיביו העיקריים (ALU, אוגרים למטרות כלליות וצובר, מונה תכנית, אוגר הוראות) שבעזרתם הוא מבצע את תפקידיו
- הזיכרון – הזיכרון כאוסף של תאים שניתן לאחסן בהם נתונים והוראות, התא, כתובת התא וערך מאוחסן בתא; יחידות זיכרון בסיסיות – בית, מילה; פעולות על תאים המאוחסנים בזיכרון – אחסון ועדכון מידע, אחזור מידע (חשוב להדגיש כי המידע מאוחסן בזיכרון בצורה אחידה, בינרית, וכי היע"מ מפרש אותו ו'מטפל' בו); סוגי זיכרונות – ראשי ומשני – והשימושים בהם; נפחי זיכרון שונים – KB , MB , GB
- יחידות קלט ופלט – הממשק המחבר בין המשתמש, המתכנת והמחשב; מגוון יחידות הקלט והפלט, למשל טקסט ממקלדת, הצבעה באמצעות עכבר, סריקת מסמכים, קלט ממחשבים אחרים המשתתפים בתקשורת, וכן יחידות פלט כגון צג, מדפסת, רשם, המציגים טקסט, גרפיקה, קול וכו'
- פסים להעברת מידע בין יחידות שונות – פסי בקרה, פסי נתונים ופסי כתובות
- אופן הביצוע של הוראה במחשב וסוג המידע הזורם בין היחידות השונות – מחזור ביצוע הוראת קריאה ומחזור ביצוע הוראת כתיבה
- סכמת מלבנים של מחשב המבוסס על מיקרומעבד 8086 (המבנה העקרוני, סוגי האוגרים, מבנה הזיכרון – סגמנטים)

פרק 4: מבוא לשפת סף

פרק זה הוא פרק המבוא לחלק השני של הקורס העוסק בתוכנה.

מטרות הפרק

- להציג את ההבדלים, את היתרונות ואת החסרונות של שפות תוכנה עיליות לעומת שפת מכונה ושפת סף מבחינת המתכנת, תוך כדי התייחסות למושגים כמו אבסטרקציה ושקיפות; להדגיש ששפת מכונה ושפת סף מתייחסות למרכיבים פיזיים ופונות ישירות ליחידות המחשב בעוד שפה עילית מתייחסת למרכיבים לוגיים כמו משתנים, הוראות בקרה, לולאות, מבני נתונים, כל זאת בלי להתייחס למימושם במחשב
- לכתוב תכנית בסיסית (הכוללת הוראות השמה וחישוב) בשפת סף למיקרומעבד 8086
- להכיר את סביבת העבודה של טורבו אסמבלר, להריץ תכניות אסמבלי מוכנות ולכתוב תכניות פשוטות

פירוט התכנים

- מרכיבי ההוראה – הפעולה והנתונים בשפה עילית ובשפת סף
- דורות של שפות תכנות – שפת מכונה, שפת סף, שפה עילית
- שפת מכונה – דוגמה להוראה בשפת מכונה, החסרונות והיתרונות של כתיבה בשפת מכונה
- שפת סף – תפקידה, היתרונות והחסרונות של השימוש בשפת סף לעומת שפת מכונה ולעומת שפות עיליות
- מבנה ההוראות בשפת סף – דוגמה להוראה פשוטה בשפת סף, למשל הוראה לחיבור שני אוגרים
- מבנה תכנית בשפת אסמבלי – מרכיבים של שורת הוראה
- כתיבת תכנית אסמבלי בסיסית בשפת סף של מיקרומעבד 8086, תוך כדי התייחסות להוראות האלה: הוראות להעברת נתונים (השמה) בין אוגרים; הוראות אריתמטיות – חיבור, חיסור, כפל וחילוק; הוראות לוגיות, הזזה וסיבוב

פרק 5: תכנות מתקדם בשפת סף של המיקרומעבד 8086

מטרות הפרק

- להציג מנגנוני תכנות מתקדמים: לולאות, שגרות, פסיקות ומחרוזות; להציג שיטות מיעון המאפשרות ביצוע של הוראות השמה וחישוב בנתונים המאוחסנים בזיכרון; לתאר את אופן הפיתוח והביצוע של תכניות במחשב ולהתנסות בכתיבת תכניות, בהרצתן ובניפוי שגיאות שיש בהן
- להרחיב את השימוש בטורבו אסמבלר; לתאר מושגים הקשורים למבנה הזיכרון, למקטעים, למבנה תכנית בשפת סף (אסמבלר ואסמבלי); להתנסות בפיתוח ובכתיבה של תכניות בסביבה זו וללמוד טכניקות וכלים לניפוי שגיאות

פירוט התכנים

- שימוש בזיכרון ובשיטות מיעון
- הוראות הסתעפות (תנאי ולולאה) – קפיצה לא-מותנית וקפיצה מותנית
- שגרות ושימוש במחסנית
- פסיקות תוכנה
- הוראות קלט ופלט
- הוראות מחרוזת
- הרצה וניפוי של תכניות – שלבי פיתוח תכנית בשפת אסמבלר; שלבי ביצוע תכנית – קומפילציה, קישור, טעינה, הרצה עד לקבלת הפלט; בדיקה וניפוי של שגיאות

פרק 6: ההתפתחות של מחשבים מודרניים

מטרות הפרק

- להציג את ההתפתחויות ואת השינויים שחלו במהלך השנים במבנה המחשבים ובחומרה שלהם ואת השינויים שתומכים בתוכנה ומאפשרים להריץ אפליקציות גדולות ומורכבות בד בבד עם צמצום זמן העיבוד; להדגיש שהתפיסה המבוססת על von Neumann לא השתנתה, ושהשיפורים בביצועים מושגים הודות לשיפורים במהירויות העיבוד ובגודל הזיכרון
- לסקור ולהכיר את משפחות המעבדים של אינטל ולהציג נקודות ציון המתייחסות לשיפורים בחומרה; להציג חלק מדורות המעבדים שבהם חל שינוי בתפיסה או חל שיפור משמעותי לעומת הדור הקודם; לפרט – לגבי כל משפחה – מאפיינים כגון מהירות עיבוד, גודל מילה וזיכרון

פירוט התכנים

- סוגי מחשבים ומושגים: מיקרומחשבים, מחשבים גדולים, מחשבי-על, מחשב מקבילי (הכולל כמה מעבדים)
- סקירה על ההתפתחות של משפחות המעבדים מתוצרת אינטל:
- 8086 – חזרה והעמקה – סכמת מלבנים, מבנה המעבד וסוגי האוגרים, פס נתונים של 16 ביט, ארכיטקטורת pipelined (לביצוע הוראות משני שלבים: BIU, EU), מעבדי עזר לשיפור ביצועי המערכת (מעבד נתונים מספריים, מעבד קלט/פלט)
- 80486 – מבנה המעבד וסוגי האוגרים, פס נתונים של 32 ביט, שיפור בשימוש במעבד מתמטי, אופני עבודה (מצב אמיתי Real Mode, מצב מוגן Protected Mode), ארכיטקטורת pipelined בת חמישה שלבים, זיכרון מטמון (כחלק ממעבד), ארגון הזיכרון הווירטואלי, מקטעים ודפים, ניהול המעבר מיישום אחד למשנהו (time switching), הרשאות (זכויות) גישה למקטעי תכנית ונתונים, הגנה על התקני קלט ופלט המשותפים ליישומים
- פנטיום – סכמת מלבנים של המעבד; השיפורים שחלו במעבד זה לעומת 40486 – פס נתונים של 64 ביט, שתי יחידות ביצוע מקבילות (v-pipeline ו-u-pipeline), יחידה לחיזוי כתובת קפיצה (branch prediction)

- פנטיום פרו – סכמת מלבנים של המעבד, הגברת מהירות המעבד באמצעות הגדלת מספר הטרנזיסטורים, הגברת מהירות השעון והגדלת מספר ההוראות למחזור שעון, זיכרון מטמון ברמה 2 מובנה, ביצוע דינמי (dyanmic execution) שנועד להתגבר על SUPER pipelining – 'צוואר הבקבוק' של מודל von Neumann.