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 שמספריהם 602 ו-603.

6. תכנית הלימודים במדעי המחשב מובאת בדפים שלפניכם, וכן אתרי 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

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.

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
- 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

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.

פרק 1: מבוא

יעדים

- הכרה ראשונית (בפרט לנבחנים שלא למדו מדעי המחשב בחטיבת הביניים) של תחום מדעי המחשב והשפעתו על תחומי ידע אחרים
- חשיפה ראשונית לחשיבה אלגוריתמית ולכתיבת תכניות
- הכרה ראשונית של מושג העצם ושל תכנות מבוסס עצמים

תכנים

- הדגמת חשיבותו של מקצוע מדעי המחשב על-ידי דיון באתגרים חישוביים מתחומי ידע שונים
- הכרת משימות חישוביות פשוטות: ניתוח המשימה, ניסוח אלגוריתמי של פתרון אפשרי
- הכרת המושג שפת תכנות
- הכרת מושג התכנית: קריאה, כתיבה, הרצה, בדיקה ותיקון תכניות פשוטות
- מחלקה ופעולת main כמסגרת בסיסית לכתיבת תכנית
- הכרת מושג העצם
- קריאה והבנה של ממשק פשוט של מחלקה קיימת, לצורך יצירת עצמים וזימון פעולות (methods) על עצמים
- יצירת עצמים באמצעות פקודת new
- זימון וביצוע פעולות על עצמים

מטרות ביצועיות[1]

- הנבחן יקרא תכנית פשוטה ויסביר את דרך פעולתה במילים שלו.
- הנבחן יקבל משימה מילולית פשוטה; הנבחן יתכנן ויכתוב אלגוריתם שפותר אותה בשפה כללית (כלומר סיפור השתלשלות האלגוריתם).
- הנבחן יממש את האלגוריתם על-ידי כתיבת תכנית והרצתה.
- הנבחן יתאר את המבנה הבסיסי של תכנית: מחלקה, פעולת main.
- הנבחן יכתוב תכנית שיוצרת ומפעילה עצמים פשוטים על-ידי תכנות בעזרת ממשק מחלקה נתון.
- הנבחן יפעיל פעולות בסיסיות של סביבת הפיתוח שמשתמשים בה בתכנית הלימודים – java או c# – ויוכל לפתח ולהריץ בה תכנית פשוטה (פתיחת קובץ תכנית, שמירת תכנית, עריכה, הרצת פקודה אחר פקודה).

הערה: מכאן ואילך, כאשר כתוב "הנבחן יממש", הכוונה "הנבחן יכתוב, יתקן ויריץ תכנית מחשב"..

פרק 2: מושגי יסוד בתכנות

יעדים

- הכרת מושגי יסוד בתכנות: משתנים, טיפוסי נתונים, ביטויים חשבוניים, קלט / פלט, חלוקת קוד לפעולות עזר
- המשך עבודה עם עצמים ממחלקות מוכנות

תכנים

- אלגוריתם
- הכרת המושג משתנה
- טיפוסי נתונים בסיסיים: שלם (int), ועשרוני (double)
- תחומי הערכים של טיפוסי הנתונים הנזכרים לעיל
- פקודות השמה
- הגדרה ואתחול של משתנים
- אופרטורים חשבוניים: חיבור, חיסור, כפל, חילוק, שארית
- פונקציות מתמטיות בסיסיות: abs, max, min, pow, random, round, sqrt
- ביטויים חשבוניים: סדר קדימויות האופרטורים ותפקיד הסוגריים
- המרה בסיסית בין טיפוסי הנתונים int ו-double
- פעולות ככלי לעידון ולחלוקה של משימות
- הגדרה וכתיבה של פעולת עזר (private method)
- הגדרת פרמטרים, והעברה לפי ערך (call by value)
- ערך מאוחזר (return value)

[1] הנושאים שמופיעים בסעיף "מטרות ביצועיות" מתארים שאלות אפשריות בבחינות.

- שגיאות לוגיות, שגיאות תחביר, שגיאות בזמן ריצה
- פעולות קלט / פלט פשוטות
- טבלת מעקב

מטרות ביצועיות

- הנבחן יסביר את המושג "אלגוריתם" ואת הקשר שלו לכתיבת תכנית.
- הנבחן יגדיר את סוגי המשתנים ואת שמות המשתנים שמתאימים למשימה מילולית נתונה.
- הנבחן יסביר מהו משתנה וכיצד הוא מיוצג בזיכרון המחשב.
- הנבחן ינסח ויעקוב אחר פעולות השמה.
- הנבחן יסביר את הצורך בטיפוסי הנתונים int ו-double ואת השוני ביניהם.
- הנבחן יחשב את הערכים של ביטויים חשבוניים נתונים.
- הנבחן יתרגם תיאור משימה חישובית מילולית לביטוי חשבוני תקין.
- הנבחן יממש משימות חישוב שכוללות ניסוח ביטויים חשבוניים.
- הנבחן יסביר (באופן בסיסי בלבד) את הצורך בהמרה בין טיפוסי נתונים שונים.
- הנבחן יסביר את הצורך בחלוקת משימה מורכבת למשימות קטנות.
- הנבחן יסביר את הצורך בפעולות ככלי לעידון ולחלוקה למשימות.
- הנבחן יקבל קטע קוד מסורבל וידע לחלקו לפעולת עזר אחת או יותר.
- הנבחן ידע להסביר את המונח "העברה לפי ערך".
- הנבחן יסביר את מנגנון הזימון של פעולות: העברת פרמטרים, אחזור ערך, השתלבות הערך המאוחזר בביטוי שכולל את זימון הפעולה.
- הנבחן יעקוב אחר ערכי משתנים בזמן ריצה.

פרק 3: ביצוע מותנה

יעדים

- הבנת ביטויים בוליאניים, מושג התנאי, הצורך בביצוע מותנה, מבנה הבקרה if ותפקידם בהקשר הכללי של משימה חישובית ומימושה
- העמקת ההבנה של עבודה עם משתנים

תכנים

- טיפוס הנתונים boolean
- יחסים: שווה, שונה, גדול, קטן, גדול או שווה, קטן או שווה
- בעייתיות השימוש ביחסים == ו- =! בהקשר של ערכים עשרוניים
- פונקציות בוליאניות (not, and, or) וטבלאות האמת שלהן
- ביטויים בוליאנים פשוטים, מורכבים, וסדר הפעולות הבוליאניות
- ביצוע מותנה: if

- ביצוע מותנה: if .. else
- תקינות קלט, מסננת קלט פשוטה הכוללת תנאי בלבד

מטרות ביצועיות

- הנבחן יחשב את הערך של ביטויים בוליאניים נתונים ויבחין בסדר הפעולות הבוליאני.
- הנבחן יתרגם תיאור מצבים מילוליים לתנאים בוליאניים פשוטים ומורכבים. תנאי מורכב הוא תנאי שבנוי מכמה קשרים לוגיים עם או בלי סוגריים.
- הנבחן יתרגם משימות מותנות מילוליות לקטעי קוד שכוללים תצורות שונות של מבני if ו- if … else.
- הנבחן יגדיר או יכתוב משפט תנאי מקונן בשלוש רמות לפחות.
- הנבחן יפצל משפטי תנאי וייצור תנאי עקיף הכולל "או" "וגם" ו"שלילה".
- הנבחן ישתמש בפעולות בוליאניות המוגדרות על עצמים.
- הנבחן יכתוב פעולות בוליאניות וישתמש בערך החוזר מהן בתוך משפט if.
- הנבחן יממש מסננת קלט פשוטה.

פרק 4: ביצוע חוזר

יעדים

- הבנה ומימוש של אלגוריתמים בסיסיים לביצוע חוזר
- תרגול ביצוע חוזר ככלי לעידון אלגוריתמים
- הבחנה בין כתיבה אלגוריתמית של לולאה ובין מימושה בשפת תכנות
- הכרה בסיסית של המושגים נכונות ויעילות של אלגוריתמים
- העמקת הידע של כתיבת, תיעוד ותיקון תכניות
- שימוש בלולאות לאלגוריתמים המצריכים מנייה או צבירה

תכנים

- ביצוע חוזר
- לולאת for
- לולאת while
- מסננת קלט עקשנית
- משימות חישוב טיפוסיות: מונים, צוברים, ערכי קיצון
- תנאי סיום
- ביצוע אינסופי
- ניתוח נכונות בעזרת טבלת מעקב
- ניתוח יעילות על-ידי ספירת איטרציות
- קינון ושילוב מבני if, for, while
- תיעוד

מטרות ביצועיות

- הנבחן יזהה בעיות שפתרונן דורש ביצוע חוזר.
- הנבחן יזהה בעיות שפתרונן מבוסס על צבירה, מנייה וחישוב של ערכי קיצון (מינימום ומקסימום).
- הנבחן יבחין בין בעיה שפתרונה דורש ביצוע חוזר באורך ידוע מראש ובין בעיה שפתרונה דורש ביצוע חוזר מותנה.
- הנבחן יסביר את הצורך במבנה הבקרה for ואת דרך פעולתו.
- הנבחן יסביר את הצורך במבנה הבקרה while ואת דרך פעולתו.
- הנבחן יתרגם תיאור משימות מילוליות לאלגוריתמים שמבוססים על ביצוע חוזר.
- הנבחן יעקוב אחר קטעי קוד של ביצוע חוזר בעזרת טבלת מעקב.
- הנבחן יסביר את הצורך בתנאי הסיום של לולאה מותנית.
- הנבחן יזהה ויסביר לולאות בעלות ביצוע חוזר אינסופי.
- הנבחן יסביר במילים שלו את נכונותו של אלגוריתם נתון לביצוע חוזר.
- הנבחן יכתוב, יבדוק ויריץ תכניות שמבוססות על מבני הבקרה for, while .
- הנבחן יפתור בעיות הדורשות קינון ושילוב של לולאות מסוגים שונים.
- הנבחן יתעד את התכניות שהוא כותב. על התיעוד לכלול הסבר לקטעי קוד מורכבים כגון לולאות.

פרק 5: מבני נתונים סדרתיים

יעדים

- הכרת מערכים כאוסף לינארי של טיפוסים מאותו סוג, עבודה עם מערכים חד-ממדיים ועם מערכים דו-ממדיים
- הכרה ומימוש של אלגוריתמי חיפוש, מיון ומיזוג

תכנים

- מערכים של טיפוסי נתונים בסיסיים
- מושגי יסוד בעבודה עם מערכים: מציין (אינדקס), אורך (length), גישה (x[i])
- הגדרה ואתחול של מערכים
- אלגוריתמים טיפוסיים:
  - חיפוש סדרתי
  - חיפוש בינארי
  - מיון הכנסה
  - מיזוג
  - השוואת יעילות האלגוריתמים

- מערך של עצמים:
  - בניית מערך עצמים (השימוש בפעולה new)
- גישה לתכונה של עצם במערך דו-ממדי

מטרות ביצועיות

- הנבחן ידע להסביר את הצורך במבנה נתונים סודר.
- הנבחן יכתוב פעולות (methods) שמבצעות חישובים שונים על מערך נתון: סכום איברים, מספר איברים שמקיימים תנאי כלשהו, ערכי קיצון, ממוצע, בדיקת קיום תכונה כלשהי, וכדומה.
- הנבחן יסביר מה מתרחש מאחורי הקלעים כשהפקודה int[] x = new int[n] מתבצעת.
- הנבחן ידע לזהות חריגה מגבולות המערך ולכתוב קטעי קוד הנמנעים מחריגה כזו.
- הנבחן יכתוב פעולה שמקבלת מערך וערך כלשהו כפרמטרים ומאחזרת את מיקום הערך במערך בחיפוש סדרתי.
- הנבחן יסביר את אלגוריתם החיפוש הבינארי במערך ממוין.
- הנבחן ינתח באופן לא פורמלי את יעילות החיפוש הלינארי ואת יעילות החיפוש הבינארי במערך ממוין.
- הנבחן יכתוב אלגוריתם ויסביר במילים שלו כיצד לבצע מיון מערך חד-ממדי בשיטת מיון הכנסה.
- הנבחן יכתוב פעולה שממזגת שני מערכים ממוינים.
- הנבחן יקבל קטע קוד שכולל הגדרה, אתחול ועיבוד פשוט של מערך עצמים. הנבחן יסביר את דרך הפעולה של הקוד.
- הנבחן יקבל משימה מילולית שדורשת הגדרה, אתחול, ועיבוד פשוט של מערך של עצמים. הנבחן יממש את המשימה.
- הנבחן יגדיר מערך דו-ממדי ריבועי או מלבני.
- הנבחן יבצע פעולות גישה לתא במערך (עצם או תכונה פשוטה) דו-ממדי, תוך שימוש מורכב במציינים. פעולות הגישה יכללו:
  - פעולות על האלכסון הראשי והמשני,
  - סריקת המערך לפי שורות, לפי עמודות, ב'שבלול',
  - הפיכת מערך.
- הנבחן יקבל קטע קוד שכולל הגדרה, אתחול, ועיבוד של מערך דו-ממדי. הנבחן יסביר את דרך הפעולה של הקוד.
- הנבחן יקבל משימה מילולית שדורשת הגדרה, אתחול ועיבוד פשוט של מערך דו-ממדי. הנבחן יממש את המשימה. למשל: מציאת הערך המקסימלי במערך, חישוב ממוצע האיברים בכל שורה במערך, וכדומה.
- הנבחן ידע לעקוב בעזרת טבלת מעקב אחר פעולות הכוללות מערכים שאיבריהם משתנים במהלך הפעולה.

פרק 6: תכנות מונחה עצמים

יעדים

- מבוא לתכנות מונחה עצמים וכתיבת מחלקות

תכנים

- חזרה: יצירה והפעלה של עצמים תוך כדי שימוש בממשק מחלקה נתון
- מושגי יסוד: מחלקה כתבנית ליצירה ולהפעלה של עצמים, שימוש לעומת מימוש
- מושגי יסוד: תכונות, בנאים, פעולות (שאילתות / פקודות)
- מצב העצם / מצב מצב תקין
- הכמסה: חשיבות העיקרון ודרך מימושו באמצעות הרשאות גישה וכתיבת פעולות מתאימות
- עצמים שאינם ניתנים לשינוי (immutable objects)
- תנאי קדם / תנאי בתר
- מחלקות המשתמשות במחלקות אחרות
- מימוש מחלקה בשפת תכנות לפי ממשק נתון
- תיעוד ממשקים
- כתיבת תכניות בדיקה ידניות ואוטומטיות

מטרות ביצועיות

- הנבחן יקבל ממשק מחלקה, מימוש של המחלקה, ומשימה מילולית שדורשת יצירה והפעלה של מספר עצמים. הנבחן יממש את המשימה על-ידי כתיבת קוד שמשתמש בשירותי המחלקה.
- הנבחן יסביר במילים שלו מה מתרחש בצד המחלקה כאשר קטע הקוד שהוא כתב לעיל מתבצע.
- הנבחן יסביר את עקרון ההכמסה ויתאר כיצד הוא בא לידי מימוש בקוד מחלקה נתון.
- הנבחן יסביר אם קוד מחלקה נתון מפר את עקרון ההכמסה וכיצד הוא מפר אותו.
- הנבחן יקבל תיאור מילולי וממשקי מחלקה מתאימים ויכתוב קוד שיוצר ומפעיל עצמים שטיפוס אחת מתכונותיהם הוא מחלקה.
- הנבחן יתאר מהו מצב תקין של עצם וכיצד הוא בא לידי ביטוי בהינתן תיאור מילולי של בעיה.
- הנבחן יתאר מהם תנאי קדם ותנאי בתר וכיצד הם באים לידי ביטוי בהינתן תיאור מילולי של בעיה.
- הנבחן יתכנן ויכתוב תכניות בדיקה ידניות ואוטומטיות בהינתן ממשק מחלקה.
- הנבחן יתעד תכניות.

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 ובשאלון 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 – **מבני נתונים** (יחידת חובה)

היחידה החמישית – הנבחנים יבחרו **באחד** מבין פרקי הבחירה האלה:

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

היחידה הרביעית – מבני נתונים

היחידה עוסקת בעיקר בבניית ובעיבוד של מבני נתונים (מחסנית, תור, עץ בינארי), בשימוש מושכל בהם, ובמימושם בעזרת מערכים ורשימות מקושרות. כמו-כן נלמד להשתמש במבנים אלו כדי לייצג ולממש טיפוסי נתונים מופשטים ומורכבים.
בנוסף לכך, היחידה כוללת שני פרקי מבוא בעלי חשיבות כללית במדעי המחשב – רקורסיה ויעילות. החומר שנלמד בשני הפרקים הללו בא לידי ביטוי אינטנסיבי בניתוח ובמימוש מבני הנתונים השונים שמוצגים בהמשך היחידה.
בשאלון **שבעה** פרקי חובה.

פרקי הלימוד

פרק 1: רקורסיה

פרק 2: מבוא ליעילות

פרק 3: מחסנית

פרק 4: תור

פרק 5: רשימה מקושרת

פרק 6: מימוש מבני נתונים

פרק 7: עצים בינאריים

סביבת העבודה

שפות התכנות האפשריות ביחידה זו הן Java ו-C#.

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

פרק 1: רקורסיה

יעדים

- הכרת מושג הרקורסיה כשיטה לפתרון בעיות
- הכרת היתרונות והחסרונות של פתרונות רקורסיביים
- ייצוג בעיות ופתרונן בעזרת רקורסיה

תכנים

- הגדרה רקורסיבית, תנאי עצירה, קריאה רקורסיבית
- רקורסיית זנב, רקורסיית הלוך-חזור, רקורסיה כפולה, רקורסיה הדדית
- מעקב רקורסיבי ברמה של f(n) ו- f(n-1)
- כתיבת תכניות רקורסיביות

מטרות ביצועיות

- הנבחן יעקוב אחר תהליך רקורסיבי וירשום את סדר המעקב במונחים של f(n) ושל f(n-1).
- הנבחן יבחין בין סוגי רקורסיות: זנב, הדדית, כפולה, הלוך-חזור.
- הנבחן יכתוב תכניות רקורסיביות פשוטות שפועלות על מספרים, סדרות, מערכים חד-ממדיים ומערכים דו-ממדיים.
- הנבחן יסביר את יתרונות הרקורסיה: פשטות הפתרון, הוכחת נכונות.
- הנבחן יסביר את חסרונות הרקורסיה: צריכת משאבי מקום וזמן.
- הנבחן יסביר מדוע אין בעיה הניתנת רק לפתרון רקורסיבי.
- הנבחן יפגין הבנה של מבנים רקורסיביים, ויפתור בעיות על מבנים רקורסיביים.

פרק 2: מבוא ליעילות

יעדים

לפרק זה שני יעדים: (א) חזרה וריענון של מערכים ואלגוריתמי חיפוש ומיון – חומר שנלמד בפרקי היסודות (י״ל 1–2); (ב) הכרת מושג היעילות וחשיבותו במדעי המחשב, וזאת דרך הכרת מדדי זמן-ריצה. בפרט, יש להכיר את מושגי היעילות הבסיסיים ״המקרה הגרוע ביותר״, ״המקרה הטוב ביותר״, ו״המקרה הממוצע״. כמו-כן יש להכיר את מושג אורך הקלט ואת מושג היעילות כפונקציה של אורך הקלט, ואת המושג ״סדר גודל״ (O גדול). יש ללמוד את המושגים באופן אינטואיטיבי, בלי להיכנס להגדרות מתמטיות. יש לנתח את יעילותם של אלגוריתמים שונים שנלמדו כבר בפרקי היסודות, כגון מציאת מקסימום, חיפוש סדרתי, חיפוש בינארי, מיון הכנסה ומיון-מיזוג.

מטרות ביצועיות

- הנבחן יפגין הבנה של תלות זמן הריצה של אלגוריתם נתון כתלות של אורך הקלט.
- בהינתן אלגוריתם כלשהו וקלטים שונים, הנבחן יזהה את המקרה הגרוע ביותר והמקרה הטוב ביותר במונחי זמן-ריצה.
- הנבחן ינתח את זמן הריצה של אלגוריתם מילולי נתון כתלות בפעולות בסיסיות ובאורך הקלט.
- הנבחן ינתח את זמני הריצה של אלגוריתמים שונים שנלמדים בתכנית הלימודים, הן בפרקי היסודות והן ביחידת הלימוד הנוכחית.

פרק 3: מחסנית

יעדים

- הכרת המושג ״טיפוס נתונים מופשט״
- הכרת טיפוס הנתונים המופשט ״מחסנית״ (stack)
- הכרת שימושים נפוצים במחסנית
- תרגול עבודה מול ממשק מחלקה (API)
- הכרת מושג הגנריות והשימוש בו

תכנים

- טיפוס נתונים מופשט
- מחסנית, LIFO
- ממשק מחלקה: בנאים, פעולות שמחזירות ערך, פעולות עדכון
- גנריות

מטרות ביצועיות

- הנבחן יסביר מהו טיפוס נתונים מופשט.
- הנבחן יסביר את טיפוס הנתונים המופשט "מחסנית" (Stack) ואת שימושיו.
- הנבחן יבנה ויעבד מחסנית כדי לפתור בעיות המאופיינות על-ידי תבנית LIFO.
- הנבחן יסביר את הממשק של מחלקת Stack.
- הנבחן יפגין יכולת לכתוב קוד שמשתמש כהלכה בשירותי ממשק מחלקת Stack.
- הנבחן יסביר את מושג הגנריות.
- הנבחן יפגין יכולת לכתוב קוד שמשתמש כהלכה בשירותי ממשק מחלקת Stack<T>.
- הנבחן יסביר כיצד ניתן לפעול על טיפוס נתונים מופשט בלי לדעת כיצד הוא ממומש.

פרק 4: תור

יעדים

- הכרת טיפוס הנתונים המופשט "תור" (queue)
- הכרת שימושים נפוצים בתור
- תרגול עבודה מול ממשק מחלקת Queue נתון

תכנים

- תור, FIFO

מטרות ביצועיות

- הנבחן יסביר את טיפוס הנתונים המופשט הגנרי "תור" ואת שימושיו.
- הנבחן ישתמש בתור כדי לפתור בעיות המאופיינות על-ידי תבנית FIFO.
- הנבחן יסביר את משמעות הממשק Queue<T> ואת דרכי העבודה מולו.
- הנבחן יבדיל בין בעיות שפתרונן מצריך מחסנית לבעיות שפתרונן מצריך תור.

פרק 5: רשימה מקושרת

יעדים

- הכרת מבנה הנתונים "רשימה מקושרת" (linked list) או "רשימה" בקיצור, ושימושיו הרבים
- מימוש רשימות ואלגוריתמים על רשימות

תכנים

- המושגים מצביע, הפניה, חוליה ורשימה
- בנייה ועיבוד של רשימות
- מחלקת Node ומחלקת BinNode
- הקצאת זיכרון סטטית ודינמית

מטרות ביצועיות

- הנבחן יסביר וידגים את המושגים מצביע, הפניה, חוליה ורשימה.
- הנבחן יקבל תרשים של רשימה ויזהה בו דוגמאות למצביע, להפניה, לחוליה ולרשימה.
- הנבחן יסביר באופן מדויק מה מתחולל מאחורי הקלעים במהלך ביצוע פעולת new.
- הנבחן יממש אלגוריתמים איטרטיביים ורקורסיביים לעיבוד רשימות: מעבר על רשימה, הוספה, ביטול וחיפוש חוליה לפי ערך או לפי מקום.
- הנבחן ינתח את יעילות האלגוריתמים הללו.
- הנבחן יסביר את היתרונות ואת החסרונות של עבודה עם רשימה לעומת עבודה עם מערך.
- הנבחן יבין את הצורך בהגדרת מחלקה שאחת (או יותר) מתכונותיה היא מטיפוס המחלקה.
- הנבחן ישתמש בממשקים של מחלקת Node ושל מחלקת BinNode כדי לממש ולעבד רשימות.
- הנבחן יסביר את ההבדלים בין הקצאת זיכרון סטטית (מערכים) להקצאת זיכרון דינמית (רשימות).

פרק 6: מימוש מבני נתונים

יעדים

בפרקים הקודמים עסקנו בטיפוסי נתונים מופשטים (ADT's) וברשימות מקושרות. בפרק זה "נפתח את הקופסה השחורה" ונלמד כיצד ניתן לממש טיפוסי נתונים מופשטים בדרכים שונות, בפרט באמצעות מערכים ורשימות. כמו-כן נממש מבני נתונים מורכבים, כגון מערך של רשימות.

תכנים

- מימוש של טיפוסי נתונים מופשטים
- הבנה כי להפשטה נתונה יכולים להיות מימושים אלטרנטיביים, ללא שינוי ההפשטה
- מימוש הטיפוסים תור ומחסנית באמצעות מערך
- מימוש הטיפוסים תור ומחסנית באמצעות רשימה
- יעילות המימושים השונים

הערה: בפרק זה, כמו בפרק הקודם, כשאנו אומרים "מימוש באמצעות רשימה" הכוונה לרשימה המיוצגת על-ידי שרשור חוליות שהן מופעים של מחלקת Node. אין צורך להשתמש בשום מחלקה נוספת לייצוג הרשימה.

מטרות ביצועיות

- הנבחן יסביר את ההבדלים בין שימוש בטיפוס נתונים מופשט ובין מימושו.
- הנבחן יסביר כיצד ניתן לממש טיפוס נתונים מופשט בדרכים שונות.
- הנבחן יממש מחסנית ותור באמצעות מערכים ורשימות (שני מימושים אלטרנטיביים).
- הנבחן יסביר את היתרונות והחסרונות של מימושים שונים.
- הנבחן יממש תור באמצעות רשימה.
- הנבחן יממש טיפוסי נתונים מורכבים כגון מערך של רשימות ומערך של תורים.

פרק 7: עצים בינאריים

יעדים

- הכרת עצים בינאריים: בנייה, שימושים ואלגוריתמי סריקה רלבנטיים

תכנים

- המבנה והמינוח של עצים
- מושג החוליה הבינארית כאבן בנייה בסיסית של עצים בינאריים
- אלגוריתמי סריקה של עצים בינאריים
- עץ חיפוש בינארי: מבנה הנתונים ואלגוריתמים רלבנטיים
- ניתוח יעילות האלגוריתמים

מטרות ביצועיות

- הנבחן יקבל תרשים של עץ וידע לזהות בו רכיבים שונים כגון שורש, עלה, מסלול וכדומה.
- הנבחן יקבל תרשים של עץ ויכתוב קטע קוד שבונה את העץ על-ידי יצירה ושרשור של חוליות בינאריות, תוך שימוש בממשק מחלקת BinNode נתונה.
- הנבחן יסביר איך ניתן להשתמש בעץ בינארי כדי לייצג ולחשב ביטוי חשבוני נתון.
- הנבחן יסביר את המושגים pre-order, in-order, post-order.
- הנבחן יממש אלגוריתמי סריקה שונים של עצים: סריקת עומק (depth first) וסריקת רוחב (breadth first). האלגוריתמים ימומשו הן באופן איטרטיבי והן באופן רקורסיבי.
- בהינתן עץ בינארי ואלגוריתם שפועל עליו, הנבחן ינתח את יעילות האלגוריתם.
- הנבחן יסביר מהו עץ חיפוש בינרי.
- הנבחן יממש אלגוריתמי חיפוש והוספה בעץ חיפוש בינארי, וינתח את יעילותם.

ממשקי המחלקות המופיעות ביחידה הרביעית

ממשק המחלקה Stack (מחסנית)

מחלקת StackInt המחלקה מייצגת מחסנית של ערכי int. מחסנית מאופיינת על-ידי נקודת הכנסה והוצאה יחידה שמשרה סדר הכנסה/הוצאה LIFO.	
בנאי	
• בונה מחסנית ריקה.	Stack()
שאילתות	
• האם המחסנית ריקה?	boolean isEmpty()
• מחזיר תיאור של המחסנית כ- [x1, x2, …, xn]. הערך שנוסף לאחרונה למחסנית מופיע ראשון.	String toString()
פקודות	
• דחיפה: מוסיף את הערך x לראש המחסנית.	void push(int x)
• שליפה: מחזיר את הערך שנמצא בראש המחסנית, ומסיר אותו מהמחסנית. תנאי קדם: המחסנית אינה ריקה.	int pop()
• הצצה: מחזיר את הערך שנמצא בראש המחסנית בלי לשנות את מצב המחסנית. תנאי קדם: המחסנית אינה ריקה.	int top()

מחלקת Stack<T> המחלקה מייצגת מחסנית של ערכים מטיפוס גנרי T. מחסנית מאופיינת על-ידי נקודת הכנסה והוצאה יחידה שמשרה סדר הכנסה/הוצאה LIFO.	
בנאי	
• בונה מחסנית ריקה.	Stack()
שאילתות	
• האם המחסנית ריקה?	boolean isEmpty()

String toString()	• מחזיר תיאור של המחסנית כ- [x1, x2, …, xn]. הערך שנוסף לאחרונה למחסנית מופיע ראשון.
	פקודות
void push(T x)	• דחיפה: מוסיף את x לראש המחסנית.
T pop()	• שליפה: מחזיר את הערך שנמצא בראש המחסנית, ומסיר אותו מהמחסנית. תנאי קדם: המחסנית אינה ריקה.
T top()	• הצצה: מחזיר את הערך שנמצא בראש המחסנית בלי לשנות את מצב המחסנית. תנאי קדם: המחסנית אינה ריקה.

ממשק המחלקה Queue (תור)

מחלקת Queue<T> המחלקה מייצגת תור של ערכים מטיפוס גנרי T. תור מאופיין על-ידי נקודת כניסה ונקודת יציאה וסדר הכנסה/הוצאה FIFO.	
	בנאי
Queue()	• בונה תור ריק.
	שאילתות
boolean isEmpty()	• האם התור ריק?
T head()	• מחזיר את הערך שנמצא בראש התור בלי לשנות את מצב התור. תנאי קדם: התור אינו ריק.
String toString()	• מחזיר תיאור של התור כ- [x1, x2, …, xn]. הערך שנוסף לאחרונה לתור מופיע אחרון.
	פקודות
void insert (T x)	• מכניס את x לסוף התור.
T remove (T x)	• מחזיר את הערך x מראש התור ומוציא אותו מהתור.

ממשק המחלקה Node (חוליה)

מחלקת NodeInt מייצגת חוליה המכילה ערך מטיפוס int והפניה לחוליה או לערך null. ניתן להשתמש במחלקה זאת כדי לייצג רשימה המורכבת משרשרת של אפס או יותר חוליות.	
בנאים	
• בונה חוליה: משים (מלשון השמה) את ערך החוליה ל-x, ואת ההפניה שלה לערך null.	NodeInt (int x)
• בונה חוליה: משים את ערך החוליה ל-x, ואת ההפניה שלה לחוליה next. ערכו של next יכול להיות null.	NodeInt (int x, NodeInt next)
שאילתות	
• מחזיר את ערך החוליה.	int getValue()
• מחזיר את החוליה הבאה, או null.	NodeInt getNext()
• האם יש חוליה נוספת?	boolean hasNext()
• מחזיר את ערך החוליה כמחרוזת.	String toString()
פקודות	
• משנה את ערך החוליה ל-x.	void setValue (int x)
• משנה את ההפניה לחוליה הבאה ל-next. ערכו של next יכול להיות null.	void setNext (NodeInt next)

המחלקה הגנרית Node<T> מייצגת חוליה המכילה ערך מטיפוס גנרי T והפניה לחוליה או לערך null. ניתן להשתמש במחלקה זאת כדי לייצג רשימה המורכבת משרשרת של אפס או יותר חוליות.	
בנאים	
• בונה חוליה: משים (מלשון השמה) את ערך החוליה ל-x, ואת ההפניה שלה לערך null.	Node (T x)
• בונה חוליה:	Node (T x, Node<T> next)

משים את ערך החוליה ל-x, ואת ההפניה שלה לחוליה next. ערכו של next יכול להיות null.	
שאילתות	
• מחזיר את ערך החוליה.	T getValue()
• מחזיר את החוליה הבאה, או null.	Node<T> getNext()
• האם יש חוליה נוספת?	boolean hasNext()
• מחזיר את ערך החוליה כמחרוזת.	String toString()
פקודות	
• משנה את ערך החוליה ל-x.	void setValue (T x)
• משנה את ההפניה לחוליה הבאה ל-next. ערכו של next יכול להיות null.	void setNext (Node<T> next)

ממשק המחלקה BinNode (חוליה בינארית)

מחלקת BinNode<T>	
מייצגת חוליה בינארית מטיפוס BinNode<T> שמכילה ערך מטיפוס T והפניות לשתי חוליות בינאריות. ניתן להשתמש במחלקה זאת כדי לייצג עץ בינארי המורכב מאפס או יותר חוליות בינאריות.	
בנאים	
• בונה חוליה בינארית: משים (מלשון השמה) את ערך החוליה ל-x, ואת שתי ההפניות שלה ל-null.	BinNode (T x)
• בונה חוליה בינארית: משים את ערך החוליה ל-x, ואת שתי ההפניות שלה לחוליות הבינאריות left ו-right. ערכן של כל אחת משתי ההפניות הללו יכול להיות null.	BinNode (BinNode<T> left, T x, BinNode<T> right)
שאילתות	
• מחזיר את ערך החוליה.	T getValue()
• מחזיר את החוליה השמאלית, או null.	BinNode<T> getLeft()
• מחזיר את החוליה הימנית, או null.	BinNode<T> getRight()
• האם יש חוליה מימין?	
• האם יש חוליה משמאל?	
• מחזיר את ערך החוליה כמחרוזת.	String toString()

פקודות	
• משנה את ערך החוליה ל-x.	void setValue (T x)
• משים את ההפניה לחוליה השמאלית ל-left. ערכו של left יכול להיות null.	void setLeft (BinNode<T> left)
• משים את ההפניה לחוליה הימנית ל-right. ערכו של right יכול להיות null.	void setRight (BinNode<T> right)

נספח: רשימת הנושאים שכלולים בתכנית הלימודים

נספח זה מפרט את כל התכנים שכלולים בתכנית הלימודים.

בכל האמור להלן, המונח "תכנית" מתייחס לתכנית הלימודים במדעי המחשב.

1. טיפוסי נתונים בסיסיים הכלולים בתכנית: int, double, boolean, char ; טיפוסי נתונים שאינם כלולים בתכנית: short, long, byte, float.
2. אופרטורים אריתמטיים שכלולים בתוכנית: % ,/ ,* ,- ,+ .
3. אופרטורי הקיצור ++ ו- -- כלולים, אך רק במובן האופרטיבי שלהם, ללא התייחסות לערך שהם מחזירים. לכן לא יתקיים דיון בהבדל בין ++i ל- i++, והשימוש יהיה תמיד דרך ++i.
4. אופרטור ההשמה = כלול. האופרטורים =%, /=, *=, -=, += אינם כלולים משיקולי מב"מ.
5. האופרטורים היחסיים =<, >, <=, <, !=, == כלולים. האופרטורים הלוגיים ! ,|| ,&& כלולים. כל האופרטורים האחרים בשפה אינם כלולים משיקולי מב"מ.
6. האופרטור הטרנרי : ? אינו כלול.
7. ההמרות (int) ו- (double) כלולות. אלה ההמרות היחידות שחובה ללמוד ; המרות אחרות עוסקות בטיפוסי נתונים שאינם נכללים בתכנית.
8. מחלקת מחרוזת וטיפוס הנתונים מחרוזת (String) כלולות בתכנית הלימודים, אך מוגבלות לפעולות האלה בלבד: charAt, contains, indexOf, length.
9. שרשור מחרוזות ומספרים באמצעות האופרטור + כלול בתכנית ; הנבחנים אמורים להבין את ההמרות ל-String שנעשות ברקע.
10. קיימות שיטות שונות לקריאת קלט מהמשתמש, ותכנית הלימודים אינה מתייחסת אליהן באופן ספציפי. בבחינות הבגרות, בכל מקום שיהיה צורך בביצוע פעולת קלט, הצורך ימומש ויתועד כדלקמן:
    קריאה לפעולת קלט שקוראת ומחזירה ערך מסוג int // . . . = int x
11. טכניקת הפלט היחידה שכלולה בתכנית היא System.out.print/ln או Console.Write/ln ב- C#. טכניקות אחרות כגון NumberFormat ו- System.out.printf או string.format אינן כלולות.
12. מחלקת Math כלולה בתכנית אך מוגבלת לפונקציות האלה בלבד:
    abs, max, min, pow, random, round, sqrt.
13. מבני הבקרה if, if/else, while, for, foreach, return כלולים בתכנית. מבני הבקרה הבאים אינם כלולים בתכנית: do/while, switch, break, continue.

14. מערכים חד-ממדיים ודו-ממדיים (מלבניים בלבד) כלולים בתכנית, וכן גם מערכים של עצמים. אתחול מקוצר של מערכים (לדוגמה int[] array = {2, 3, 5, 7};) כלול. מכיוון שהתכנית מוגבלת למערכים מלבניים, אין צורך לדון בכך שמערך רב-ממדי ממומש ב-Java כמערך של מערכים או ב-C# על-ידי הגדרה פשוטה. ככלל, יש להתייחס לביטוי כמו x[2] [3] או בהתאמה x[2,3] ב- C# כאל תופעה סינטקטית בלי להתעמק בייצוג הפנימי/האובייקטלי של המערך. התכונה array.length ומבנה הבקרה foreach כלולים בתכנית ויש לעודד את השימוש בהם בעבודה עם מערכים.

15. העמסת מושכלת של בנאים (constructor overloading) והבנה של מושג חתימת פעולה (method signature) כלולים בתכנית.

16. פקודת new ושימוש מושכל בבנאים כלולים בתכנית.

17. מימוש מחלקות לפי API נתון כלול בתכנית. עיצוב מחלקות (דהיינו, תכנון ה API) כלול בתכנית רק ברמה של הבנת הנקרא. כלומר, נבחנים אמורים להבין את שיקולי המעצב אך אינם נדרשים לבצע משימות עיצוב בעצמם.

18. הרשאות גישה: כל המחלקות הכלולות בתכנית והמחלקות שהנבחנים יידרשו לכתוב הן public. כל התכונות הן private, פרט למקרים מיוחדים (כמו מחלקות "ערך") שבהן ניתן לדון באפשרות להשתמש בתכונות public. פעולות, בנאים, וקבועים הם תמיד public או private. הרשאות הגישה protected ו-package-private אינן כלולות בתכנית.

19. הערות מסוג //, /* ... */, /* ... **/, כלולות בתכנית.

20. המאפיין final כלול רק בהקשר של קבועים. המאפיין final בכל הקשר אחר (מחלקות, פעולות, פרמטרים, תכונות) אינו כלול בתכנית.

21. this כלול בתכנית, אך יש להגביל את השימוש בו לקוד של: (א) בנאים עם פרמטרים ששמותיהם זהים לשמות תכונות, ו- (ב) פעולות בהן מתעורר צורך להתייחס לפרמטר מסוג עצם של אותה מחלקה.

22. הפעולות equals, hashcode ו-clone אינן כלולות בתכנית.

23. השימוש בפקודת import והבנה כללית של מושג ה package כלולים בתכנית.

24. מחלקות מקוננות ומחלקות פנימיות אינן כלולות בתכנית.

25. threads אינם כלולים בתכנית.

26. החריגות הטיפוסיות שקורות בזמן הרצת התכניות (NullPointerException, ArrayIndexOutOfBoundsException, ArithmeticException, IllegalArgumentException) כלולות בתכנית. המבנים try / catch / finally ו-throws אינם כלולים בתכנית.

27. ירושה, ממשקים (interface), מחלקות אבסטרקטיות ופולימורפיזם אינם כלולים בתכנית. יש להכיר את המושגים ירושה וממשקים (interface) כדי להבין את משמעותם בקריאת תיעוד API של מחלקות שונות. יחד עם זאת, כאמור, השימוש בהם אינו כלול בתכנית.

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

- זקס ש' (1991), **אוטומטים ושפות פורמליות**, כרכים א' ו-ב', האוניברסיטה הפתוחה.
- הראל ד' (1991), אלגוריתמיקה, **יסודות מדעי המחשב**, האוניברסיטה הפתוחה.
- Barwise, Etchemendy (1993), **Turing's World, An Introduction to Computability Theory**, CLSI Publications.
- Davis, Sigal, Weyuker (1994), **Computability, Complexity and Languages, Fundamentals of Theoretical Computer Science**, Academic Press.
- 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 מוקדשת 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

יעדים

- הכרת המושגים הבסיסיים בתחום

תכנים

- 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

יעדים

- הצגת מודל האוטומט הסופי הדטרמיניסטי
- תרגול בניית אוטומטים

תכנים

- 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

יעדים

- הכרת מושגים בסיסיים בתורת השפות הפורמליות
- חקירת כוחו של מודל האוטומט הסופי הדטרמיניסטי והתכונות של משפחת השפות הרגולריות

תכנים

- 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

יעדים

- הבנה כיצד אפשר – על-ידי שינוי הגדרה קיימת של מודל חישובי – לקבל מודלים חדשים
- הכרת מושג האי-דטרמיניזם ודיון בהשוואת כוחם של מודלים חישוביים

תכנים

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

יעדים

- הכרת מודל אוטומט המחסנית הלא-דטרמיניסטי
- תרגול בניית אוטומט המחסנית

תכנים

שימוש במחסנית כמבנה עזר, אוטומט מחסנית לא-דטרמיניסטי

פרק 6: כוחו ומגבלותיו של מודל אוטומט המחסנית

יעדים

- הכרת כוחו ומגבלותיו של מודל אוטומט המחסנית
- השוואה בין המודל החדש למודל האוטומט הסופי

תכנים

- אוטומט מחסנית דטרמיניסטי
- השוואה בין כוח החישוב של אוטומט מחסנית לא-דטרמיניסטי ובין 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

יעדים

- הצגת מכונת טיורינג כמודל לתכנית מחשב
- הכרת התזה של צ'רץ' וטיורינג

תכנים

- 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 Java ו-C#

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 Java. אולם כיוון שרוב ההבדלים בין Java ל-C# הם תחביריים, ומכיוון שבמגמת הנדסת תוכנה אפשר ללמוד Web Services בכל שפות ה- NET כל האמור לגבי Java 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 ב-Java

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 ב-Java. בסביבת העבודה ניתן גם להריץ את התכניות.

סביבת העבודה ב-C#

סביבת העבודה היא סביבת NET. בכל פעם שתכנית הלימודים מתייחסת לתחביר Java 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 Java. 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 Java. 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,

  (מאמר העוסק בהבדלים שבין Gava ל-C#)

פרקי הלימוד

פרק 1: כל העולם כולו עצמים

פרק 2: עוברים ל-Java

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 Java

פרק הרחבה: מחלקות מופשטות

פרק 1: כל העולם כולו עצמים

יעדים

- הצגה של מושגי היסוד ושל עקרונות הגישה 'תכנות מונחה העצמים'
- שימת דגש על מגוון רחב של דוגמאות המקשרות את הגישה לחיי היום-יום של הנבחנים, ולידע קודם מלימוד מדעי המחשב

תכנים

- 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 ל-Java

יעדים

- הקניית היכולת לתכנת בשפת Java ברמה מקבילה כמעט ליכולת התכנות שהייתה להם בשפה הפרוצדורלית
- לימוד כללי הכתיבה (המוסכמות) בשפת Java
- לימוד מבני בקרה והקניית שליטה בזרימת התכנית של שפת Java
- הכרת סביבת העבודה –JCreator , והתנסות בה

תכנים

- דקדוק, תחביר ומוסכמות: נקודה-פסיק, סימון בלוק פקודות, הערות, הזחה (Indent), מילים שמורות ראשונות; מוסכמות כתיבה ראשונות ב-Java (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

יעדים

- חזרה על המושגים היסודיים שנלמדו בפרק המבוא, תוך כדי מימושם בשפת Java
- שימוש ראשוני במושגי המחלקה, העצם והתקשורת בין עצמים
- כתיבת תכניות ראשונות ומחלקות ראשונות

תכנים

- 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

יעדים

- סקירת נושאים תחביריים מתקדמים (מערכים, מחרוזות ואיברי מחלקה)
- העמקה בנושא הרשאות גישה
- הכרת מנגנון התיעוד של Java (Javadoc)

תכנים

- 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

יעדים

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

תכנים

- ירושה – מייצגת חלוקה היררכית טבעית-אנושית : תהליך של מיון וסיווג היררכי ; ייצוג היחס 'סוג של' (is a)
- ירושה ב-Java (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 ב-Java, תוך כדי הדגשה של צמצום השימוש בו לטובת רעיון הפולימורפיזם
- מחלקות עוטפות – הדרך להפוך טיפוסים בסיסיים לאובייקטים

פרק 6: ממשקים

יעדים

- הכרת מנגנון הממשקים (interfaces) ב-Java
- הבנה כיצד המנגנון מאפשר שימוש מתקדם בעקרון הפולימורפיזם

תכנים

- 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 ב-Java: 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 Java

יעדים

- הצגת הרקע ההיסטורי והמחקרי להתפתחות של גישת התכנות מונחית העצמים
- הבנת הסדר הכרונולוגי של פיתוח שפות תכנות
- הבנת הרציונל והמטרות שעמדו בפני מפתחי שפות התכנות

הערה: זהו פרק העשרה בלבד

תכנים

- 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
- Java 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

יעדים

- הכרת המנגנון של מחלקות מופשטות (abstract classes) ב-Java
- הבנה כיצד המנגנון מאפשר שימוש מתקדם והרחבה של עקרון הפולימורפיזם

תכנים

- הצורך בהגדרת מחלקות מופשטות: המחלקה המופשטת כמגדירה רעיון שאינו ניתן למימוש וליצירה של עצמים בשלב נתון
- המחלקה המופשטת ב-Java: 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 האלה: 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://weierstrass.is.tokushima-u.ac.jp/ikeda/suuri/simplex/Simplex.shtml

  - מילון מונחים בתורת הגרפים:

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

  - אתרי קורסים המכילים חומר לימודי (class notes):

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

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

  - פורטל לחקר ביצועים:

http://mat.gsia.cmu.edu

פרק 1: מודל התכנון הלינארי

יעדים

- הכרת ההנחות והמושגים הבסיסיים של מודל התכנון הלינארי
- תרגול של ניסוח בעיות בעזרת מודל התכנון הלינארי

תכנים

- 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

יעדים

- הצגה ותרגול של הפתרון הגרפי של בעיית תכנון לינארי בעלת שני משתנים
- חקירת התכונות העיקריות של בעיות תכנון לינארי בעלות שני משתנים
- הצגת ההשלכות של הפתרון הגרפי על התכונות של בעיות תכנון לינארי בעלות N משתנים
- הכרת העקרונות הבסיסיים של שיטת הסימפלקס
- הדגמת השלבים השונים בשיטת הסימפלקס

תכנים

- פתרון גרפי של בעיית תכנון לינארי בעלת שני משתנים; תיאור גרפי של התחום האפשרי במקרים האלה: 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

יעדים

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

תכנים

- 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

יעדים

- הדגמת מודלים של בעיות זרימה ברשתות והכרת המושגים הבסיסיים של בעיות אלו
- הצגת הגרף כטיפוס נתונים מופשט
- הדגמת אלגוריתמים של בעיות זרימה ברשתות

תכנים

- 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

יעדים

- הצגת בעיות המסלול הקצר ביותר והכרת המושגים הבסיסיים של בעיות אלו
- הצגה וניתוח היעילות של שיטות שונות לפתרון בעיות המסלול הקצר ביותר
- תרגול ניסוח ופתרון של בעיות המסלול הקצר ביותר

תכנים

- 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

יעדים

- הכרת המושגים הבסיסיים של בעיית העץ הפורש
- הצגה וניתוח היעילות של שיטות שונות לפתרון בעיית העץ הפורש

תכנים

- 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

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

- 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:

- היא מאפשרת הבנה טובה יותר של מבנה המחשב.
- היא מזמנת לנבחן היכרות עם שפת תכנות המתאימה לביצוע יישומים שבהם הזמן הוא קריטי (יישומי זמן אמת).
- 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

יעדים

הכרת 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.

תכנים

- 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

יעדים

- לימוד האופן שבו מאוחסן במחשב מידע מסוגים שונים (הוראות תכנית, נתונים, בקרה וכו') והסיבות לכך שהמידע מאוחסן במחשב בשיטת ספירה אשר שונה משיטת הספירה העשרונית
- לימוד 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

תכנים

- סוגי מידע שמאוחסנים וזורמים במחשב – הוראות (של תכנית), נתונים, הוראות בקרה, כתובות
- שיטות ספירה (ייצוג) – עשרונית, בינרית והקסהדצימלית; סדרי גודל של מספרים ביניים – GB , MB , kB
- 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

יעדים

- הצגת היחידות הבסיסיות של המחשב והמבנה של כל אחת מהן כדי להבין כיצד תכנית מתבצעת בצורה מפורטת
- הצגת המרכיבים הבסיסיים והתפקידים של כל יחידה, אשר מאפשרים לבצע תכנית במחשב
- הכרת מושגים חשובים ולימוד מהי שפת סף ומהו המבנה של מיקרו-מחשב 8086
- התייחסות לרכיבים כאל קופסאות שחורות (בסכמת מלבנים מפורטת) ולתפקידים 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.

תכנים

- 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 – GB , MB , KB
- 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.

יעדים

- הצגת ההבדלים, היתרונות והחסרונות של שפות תוכנה עיליות לעומת שפת מכונה ושפת סף מבחינת המתכנת, תוך כדי התייחסות למושגים כמו אבסטרקציה ושקיפות; הדגשה 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
- כתיבת תכנית בסיסית (הכוללת הוראות השמה וחישוב) בשפת סף למיקרומעבד 8086
- הכרת סביבת העבודה של טורבו אסמבלר, הרצת תכניות אסמבלי מוכנות וכתיבת תכניות פשוטות

תכנים

- 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

יעדים

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

תכנים

- 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 אסמלבר; שלבי ביצוע תכנית – קומפילציה, קישור, טעינה, הרצה עד לקבלת הפלט; בדיקה וניפוי של שגיאות

פרק 6: ההתפתחות של מחשבים מודרניים

יעדים

- הצגת ההתפתחויות והשינויים שחלו במהלך השנים במבנה המחשבים ובחומרה שלהם והשינויים שתומכים בתוכנה ומאפשרים להריץ אפליקציות גדולות ומורכבות בד בבד עם צמצום זמן העיבוד; הדגשה שהתפיסה המבוססת על von Neumann לא השתנתה, ושהשיפורים בביצועים מושגים הודות לשיפורים במהירויות העיבוד ובגודל הזיכרון

- סקירה והכרה של משפחות המעבדים של אינטל והצגת נקודות ציון המתייחסות לשיפורים בחומרה; הצגת חלק מדורות המעבדים שבהם חל שינוי בתפיסה או חל שיפור משמעותי לעומת הדור הקודם; פירוט – לגבי כל משפחה – של מאפיינים כגון מהירות עיבוד, גודל מילה וזיכרון

תכנים

- 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, ארכטיקטורת 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