

פרק 1

כל העולם כולו עצמים

א. מהו תכנות מונחה עצמים

ביחידה זו נלמד גישה תכנותית המכונה תכנות מונחה עצמים **Object Oriented Programming**. כדי להבין מהו הרעיון הכללי העומד בבסיס גישה זו, דמיינו לעצמכם שעליכם לתכנת מערכת תוכנה גדולה ומורכבת, כגון מערכת לניהול חשבונות בנק, או מערכת לניהול שוטף של בתי ספר. תוכלו לתכנת מערכת כזו בכל שפת מחשב עילית, כגון שפת C או פסקל. מתכנת תמים יכתוב תוכנית שמורכבת מסדרה ארוכה של פקודות, של לולאות ושל מבני בקרה. מתכנת מתקדם יותר, יחלק את הקוד המורכב לפונקציות ולפרוצדורות. מתכנת מומחה יחלק את הקוד לקבצים אחדים, וישתמש בכלים כגון יחידות הספרייה של טורבו פסקל או חלוקה לקובצי מימוש ולקובצי ממשק ב-C. כיום ידוע כי חלוקת התוכנית ליחידות עצמאיות היא תנאי הכרחי להצלחתה.

גישת התכנות מונחה העצמים (ובקיצור תמ"ע או OOP) מקדמת את רעיון החלוקה של הקוד צעד קדימה. ביסודה של גישה זו עומד הרעיון כי את המשימה הגדולה שאנו מצפים מהתוכנית לבצע ניתן לחלק למשימות קטנות. על כל משימה אחראית יחידת תוכנה המכונה "עצם" (או אובייקט). כל עצם הוא יחידה סגורה ועצמאית הממלאת תפקיד מסוים. חלקים אחרים בתוכנית יוכלו לפנות לכלל עצם שהוא ולבקש ממנו לבצע פעולה מסוימת. התוכנית השלמה היא אוסף גדול של עצמים המתקשרים ביניהם ופועלים אלו עם אלו.

כדי להתוודע ליתרונותיה של גישה זו, יש ללמוד אותה ולהכירה היטב. על רגל אחת נציין כי בגישה כזו קל יותר לתכנן ולפתח מערכות תוכנה גדולות. משום שהמערכת מורכבת מעצמים אפשר לתחזק את הקוד ביתר קלות (ניתן לתקן או לשנות יחידות מסוימות בודדות בלי שתהיה לכך השפעה על חלקים אחרים). כמו כן, קל יותר לשתף מתכנתים רבים בפיתוח משותף (כל מתכנת יכול לעבוד על חלק עצמאי). נוסף על כך, גישת תכנות מונחה עצמים שואפת להיות טבעית וקרובה לאופן החשיבה של המתכנתים על מערכות תוכנה כמו גם על מערכות אחרות בעולם.

אנו מקווים כי בסיום לימוד יחידה זו תכירו את גישת ה-OOP, ותגלו מקצת האפשרויות הטמונות בה.

אז מהו בעצם "עצם"?

המילה "עצם" משמשת אותנו רבות בחיי היום-יום, אולם בספר זה נתייחס אל עצמים בהקשר התכנותי. **עצם (object)** בתוכנית הוא יחידה סגורה ועצמאית. את העצם מגדירות **תכונות (attributes)** המייצגות את מצבו, ופעולות שאותן נכנה **שיטות (methods)** המייצגות את התנהגותו. את אוסף הרכיבים הללו – תכונות ושיטות – נכנה **איברים (members)**. תוכנית מחשב תורכב מאוסף של עצמים שיפעלו אלו עם אלו על מנת לבצע משימה משותפת.

חשבו לדוגמה על טלפון סלולרי כעל עצם. לכל טלפון סלולרי ישנן תכונות: כמות המקום בזיכרון, מצב הסוללה, רמת הקרינה הנפלטת ממנו וכן הלאה. בכל טלפון סלולרי ניתן לבצע פעולות (או שיטות) שונות: שינוי סוג הצלצול, בירור מצב הסוללה, משיכת הודעות מתא קולי וכדומה.

באופן דומה, לכל עצם בתוכנית המחשב שלנו יש תכונות שמייצגות את מצבו ושיטות שניתן להורות לו לבצע.

חשיבה בעצמים היא חשיבה טבעית

לא במקרה בחרנו בדוגמה מחיי היום-יום (טלפון סלולרי) כדי להבהיר מהו עצם בתוכנית. אחד הרעיונות המרכזיים העומדים ביסוד גישת ה-OOP הוא כי באופן טבעי אנו תופסים את העולם כמורכב מעצמים. בחיי היום-יום אנו נתקלים בעצמים רבים, שאותם אנו מנתחים לפי תכונותיהם ופעולותיהם בלי להיות ערים לכך, ובלי לציין זאת במפורש. בחינת העצם לפי התכונות והפעולות שלו היא תהליך טבעי ואינטואיטיבי בעבורנו.

נבחן לדוגמה מספר עצמים, ונזהה בכל אחד מהם את תכונותיו ואת פעולותיו. השמש היא ללא ספק אחד העצמים המרכזיים בחיינו. ברשימת תכונותיה של השמש כלולים: קוטר, מסה, גיל והרכב כימי. גורמים אלו מכונים בשם "תכונות" משום שהם מגדירים את מצבה של השמש. התנהגותה של השמש מוגדרת על ידי שורה של פעולות שהיא מבצעת, כגון פליטת קרינה, תנועה בשמים לפי מסלול, זריחה ושקיעה.

לווייתן, אף הוא עצם. בין תכונותיו נמנים: אורכו, משקלו, צבעו, גילו וכן הלאה; ובין פעולותיו: שחייה, צלילה, ציפה מעל פני המים, נפנוף בסנפיר, התזת סילון מים מן הנחיר ועוד פעולות רבות כיד הדמיון הטובה עליכם.

גם חשבון בנק הוא עצם. שלא כמו שני העצמים הקודמים, חשבון בנק הוא עצם מופשט – איננו יכולים לראות אותו או לגעת בו (משום כך, ייתכן שההתייחסות אליו כאל עצם תיראה במבט ראשון מוזרה מעט). בין תכונותיו של חשבון הבנק נימנים מספר החשבון, יתרת הכסף בחשבון, בעל החשבון והסכום המרבי המאושר למשיכת יתר. פעולותיו של עצם זה מוכרות לכולנו כ"פעולות בחשבון", וכוללות בין השאר: הפקדה, משיכה ובירור יתרה.

אין חולק על כך שגם שולחן הוא עצם. קל לשער מהן תכונותיו של עצם זה, אולם ניסיון לברר מהן פעולותיו עשוי להוביל למבוי סתום. בעתיד נראה כי לעתים אפילו לעצם דומם כשולחן ניתן לייחס פעולות מסוימות. למעשה, ניתן להתייחס לכל פריט בעולם – חי, צומח, דומם או מופשט – כאל עצם, שאותו מאפיינות תכונותיו ועל פי רוב גם פעולותיו.

מן הדוגמאות השונות עולה המסקנה כי אנו מכירים עצמים ורגילים אליהם. כאמור ההתייחסות לעולם שסביבנו כמורכב מעצמים היא טבעית ואינטואיטיבית. עובדה זו מקלה עלינו מאוד כאשר אנו כותבים תוכניות בגישת ה-OOP: תוכנית המחשב שלנו תבנה מעין עולם פנימי, המורכב מעצמים "וירטואליים" אשר יאופיינו על ידי מצב והתנהגות, בדומה לעצמים אמיתיים.

עד כאן דיברנו על עצמים ממש, כגון מכונית. בתכנות מונחה עצמים נייצג עצמים כאלה על ידי יחידות תוכנה הקרויות "עצמים". את התכונות של העצמים הממשיים, שלהם חשיבות בתוכנית, ייצגו תכונות של עצמי התוכנה. את השינויים החלים בעולם הממשי ייצגו פעולות של עצמי התוכנה.

ב. תכונות של עצם

תכונות מתארות מצב

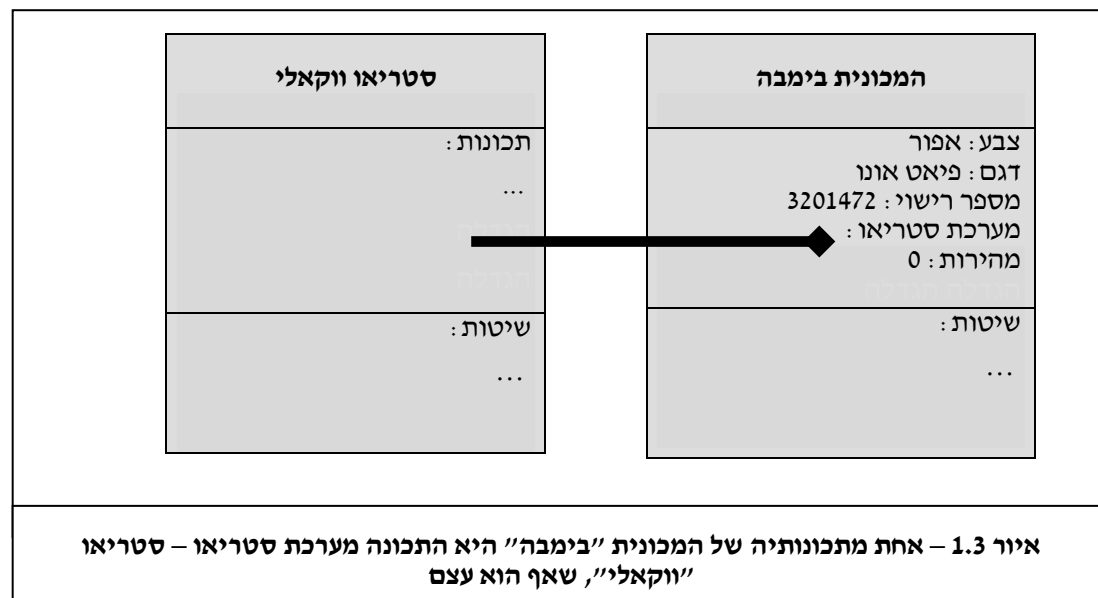
ראינו כי לכל עצם יש תכונות שקובעות את מצבו. כדי לתאר עצם מסוים אין די בציון תכונותיו, אלא יש לפרט גם אילו ערכים הן מקבלות. לדוגמה, כדי לתאר מכונית לא נמנה רק את תכונותיה: "דגם, צבע ומספר רישוי", אלא נפרט: "המכונית היא מדגם פיאת אונו, צבעה אפור ומספר הרישוי שלה הוא 32 014 72". שינוי במצב העצם יתבטא בשינוי ערכי התכונות שלו. למשל הגברת מהירותה של מכונית מתבטאת בהגדלת ערך התכונה "מהירות".





תכונות שהן עצמים

בפסקה הקודמת ציינו תכונות מטיפוסים בסיסיים (מספרים, תווים וכיוצא בזה). לעתים תכונות העצם מורכבות, וטיפוסים בסיסיים אינם מתאימים לייצג אותם. במקרים כאלה נייצג את תכונותיו של עצם באמצעות עצמים נוספים. למשל לעצם "מכונת" שייכת התכונה "מערכת סטריאו". למערכת זו תכונות משל עצמה: מצב (כבוי או דלוק), עוצמת קול, מספר תקליטורים במגש התקליטורים, מספר מגברי הקול שמחוברים למערכת ועוד כהנה וכהנה. מובן שבלתי סביר לייצג את המערכת באמצעות ערך מטיפוס בסיסי, ולכן נייצג אותה על ידי עצם נוסף: "מערכת סטריאו". במקרה זה נאמר שעצם המכונת **מורכב** בין היתר מהעצם "מערכת סטריאו". אגב, גם העצם "מערכת סטריאו" יכול להיות מורכב מעצמים אחרים. בין תכונותיו ניתן לכלול עצמים כגון: רדיו, נגן תקליטורים ועוד, על פי רמת הפירוט הנדרשת.



ג. שיטות של עצם

שיטה היא פעולה שהעצם יודע לבצע

עצמים יודעים לבצע פעולות שונות. פעולות אלו מכונות שיטות. שיטותיו של העצם המייצג מכונות הן יכולות להיות:

- עצור ()
- הדלק-אורות ()
- כבה-אורות ()
- קבע-מהירות (מהירות-מבוקשת)

כדי להדליק את אורות המכונית למשל, נבקש מהמכונית להפעיל את השיטה "הדלק-אורות)". אם נרצה לעצור את המכונית, נבקש ממנה להפעיל את השיטה "עצור)". שיטה זו מחקה את הלחיצה על דוושת הבלם במכונית אמיתית. שיטה יכולה לקבל ביטוי כפרמטר. ערך הביטוי מועבר לה לפני ביצועה. כך נוכל לשנות את מהירות המכונית על ידי הפעלת השיטה "קבע-מהירות(מהירות מבוקשת)", שמקבלת כפרמטר את המהירות המבוקשת.

שיטות יכולות לקבל ערכים

ערכה של התכונה "מהירות" מייצג את מהירותה של המכונית ברגע נתון. נוכל להשתמש בשיטה: "קבע-מהירות(מהירות-מבוקשת)" כדי להציב ערך בתכונה זו. בין הסוגריים השיטה מקבלת ביטוי שערכו המספרי מייצג את המהירות הרצויה. בכל פעם שנעביר לשיטה ערך חיובי, האובייקט "מכונית" יתאים את מהירותו לערך המבוקש. הדבר מקביל ללחיצה על דוושת הגז במכונית אמיתית. באופן דומה נוכל להתייחס לשיטות האחרות כאל כלים שבעזרתם ניתן לשנות תכונות שונות של העצם.

שיטות יכולות להחזיר ערכים

בכל מכונית יש מד דלק. הסתכלות במד הדלק היא מעין קריאה לשיטה של המכונית, שמחזירה לנו ערך: את כמות הדלק הנוכחית שבמכל הדלק של המכונית. שיטה נוספת המחזירה ערך אפשר לזהות במכשיר טלפון סלולרי. בכל טלפון סלולרי קיימת פונקציה המאפשרת לברר את מצב הטעינה של הסוללה. לחיצה על הכפתור המתאים גורמת ל"טלפון הסלולרי" להפעיל את השיטה שמחזירה את המידע על כמות האנרגיה שנותרה בסוללה. קודם ראינו כי שיטות יכולות לקבל ערכים, כעת אנו רואים כי שיטות עשויות גם להחזיר ערכים.

נשמע מוכר?

שיטות דומות במידה רבה לפונקציות או לפרוצדורות שבהן נתקלנו בעבר, אך נבדלות מהן בכך שהן מקושרות לעצם מסוים ו"מכירות" את תכונותיו.

ד. תהליך פיתוח של תוכנית מונחית עצמים

מבנה התוכנית לפי OOP

לפי גישת תכנות מונחה עצמים, השלב הראשון בפיתוח תוכנית כלשהי הוא ההחלטה אילו עצמים תכלול התוכנית, ומה יהיה תפקידו של כל עצם. עלינו לבחור עצמים שיוכלו לבצע את המשימה הדרושה תוך אינטראקציה ביניהם. בהתאם לתפקיד שכל עצם אמור לבצע, נחליט מה יהיו איבריו.

בעיה לדוגמה: תוכנה לניהול סופרמרקט

כדי להמחיש את כל האמור לעיל, נציג תהליך פיתוח של תוכנה למרכול גדול – סופרמרקט. תוכנה זו יכולה לשמש כדי לנהל את חשבונות הסופרמרקט (למשל מעקב אחר הכנסות והוצאות) או כדי לנהל את מלאי המוצרים בחנות (אילו מוצרים צריך להזמין? אילו מוצרים צריך להעביר ממחסני המלאי למדפים?).

בתוכניתנו, כצעד ראשון בפיתוח התוכנה, נרצה לאבחן את המרכיבים העיקריים בסופרמרקט, את המצבים השונים הקיימים בו ואת הפעולות המתבצעות בו. בסופרמרקט קיימים לקוחות, מוצרים וקופות. לכל מוצר יש מחיר וברקוד המייצג את קוד המחיר שלו. לכל קופה יש מספר סידורי. חלק מהמוצרים נמצאים על המדפים וחלק במחסני המלאי. ייתכן גם שהסופרמרקט מסווג למדורים שונים: ירקות ופירות, מוצרי ניקיון, מוצרי חלב וכן הלאה. הלקוחות יכולים לקנות מוצרים ולשלם בעבורם בקופות. הלקוחות יכולים גם להזדכות על מוצרים שונים בקופות. את כל המצבים הללו נרצה לייצג בתוך עולם התוכנית שלנו.

בחירת העצמים

בעולם האמיתי, בכל סופרמרקט עצמים רבים: מוצר, מדף, לקוח, קופאי, עגלת קנייה, יצרן המוצר, מחסן מלאי, צג קופה, חשבונית, רשימת מוצרי המרכול, קופה וכן הלאה. כעת עלינו להחליט אילו מביניהם נייצג כעצמים בתוכנית. את תהליך הבחירה של העצמים שיתפקדו ב"עולם התכנותי" שאותו ניצור מכנים "מידול" (modeling).

ההחלטה אילו עצמים נבחר למדל בתוכניתנו מבוססת בראש ובראשונה על הבעיה שאותה אנו מנסים לפתור, כלומר על המשימות שעל התוכנה לבצע. החלטה זו מושפעת כמובן גם מסגנון אישי ונשענת על ניסיון תכנותי רב. כדי לטפל בנושא של עריכת חשבון ללקוח למשל, נבחר למדל בתוכנית רק שני עצמים הנראים הכרחיים לפתרון: "מוצר" ו"קופה". אילו היינו רוצים להתייחס גם לניהול המלאי, היינו מוסיפים את העצמים "מדף" ו"מחסן מלאי".




הגדרת העצמים – תכונות ושיטות

לאחר שהחלטנו אילו עצמים נכלול בתוכנית, עלינו להחליט מהם האיברים המרכיבים אותם. נטפל תחילה בתכונות, ולאחר מכן בשיטות. מה יהיו תכונותיהם של העצמים "מוצר" ו"קופה"? נתחיל במוצר. אחת הדרישות שציינו היא כי "לכל מוצר יש מחיר ומספר ברקוד". מכך נסיק כי




שתי תכונות רצויות של המוצר הן: "מחיר" ו"מספר ברקוד". באופן טבעי את התכונות האלו ייצגו משתנים מספריים.

דרישה נוספת: "המרכול מסווג למדורים שונים". ניתן יהיה לספק דרישה זו אם נוסיף למוצר את התכונה "מספר מדור". כך, לאחר שנוהה את המוצר בקופה, ניתן יהיה לברר מהו ערכה של תכונה זו. גם תכונה זו תיוצג באופן טבעי על ידי משתנה מספרי שלם.

נשים לב כי למוצר (האמיתי) ישנן תכונות נוספות כגון צבע ומשקל. אולם, מתוך הנחה כי אין לנו צורך בתכונות אלו כדי להתמודד עם משימות התוכנה שאותה אנו כותבים, לא נכלול אותן במודל התכנותי של המוצר.

<table border="1"><tr><td>מוצר</td></tr><tr><td>תכונות: מספר ברקוד: 0152439 מחיר: 15.00 מספר מדור: 3</td></tr><tr><td>שיטות: </td></tr></table>	מוצר	תכונות: מספר ברקוד: 0152439 מחיר: 15.00 מספר מדור: 3	שיטות: 
מוצר			
תכונות: מספר ברקוד: 0152439 מחיר: 15.00 מספר מדור: 3			
שיטות: 			
איור 1.4 – הצגה סכמטית של תכונות העצם "מוצר". נותר להשלים את שיטותיו			

נעבור לעצם "קופה". הדרישה "לכל קופה יש מספר סידורי" תביא באופן טבעי להכללת "מספר סידורי" בין תכונות הקופה. תכונה נוספת של העצם היא "סכום הכסף שבקופה". אף על פי שתכונה זו אינה מופיעה במפורש ברשימת הדרישות, נזדקק לה כדי לאפשר תשלום בקופה.

<table border="1"><tr><td>קופה</td></tr><tr><td>תכונות: מספר סידורי: 1 סכום כסף: 7654.32</td></tr><tr><td>שיטות: </td></tr></table>	קופה	תכונות: מספר סידורי: 1 סכום כסף: 7654.32	שיטות: 
קופה			
תכונות: מספר סידורי: 1 סכום כסף: 7654.32			
שיטות: 			
איור 1.5 – תכונות ה"קופה"			

כעת, לאחר שקבענו מה יהיו תכונות העצמים, נעבור לתיאור שיטותיהם. אילו שיטות יהיו ל"קופה"? מעיון בהגדרת הבעיה עולה שהקופה צריכה לבצע פעולות הקשורות בתהליכי קניית מוצר והזדכות עליו. לצורך ביצוע משימות אלו נוסיף את שלוש הפעולות הבאות:

- תשלום עבור מוצר
- בירור מחיר של מוצר
- הזדכות על מוצר

לפיכך, נוסף שלוש שיטות מתאימות ל"קופה". כולן יקבלו כערך את המוצר, או את מספר הברקוד שלו המאפשר לזהותו. השיטה השנייה מחזירה את מחיר המוצר. הערך שיחזירו השיטות הראשונה והשלישית נתון להחלטתנו.

קופה
תכונות: מספר סידורי: 1 סכום כסף: 7654.32
שיטות: שלם(מוצר) ברר מחיר(מוצר) הזדכה(מוצר)
איור 1.6 – העצם "קופה" על כל איבריו

נחזור לשיטות של העצם האחר: המוצר. בניגוד לקופה, שפעולותיה היו ברורות, במבט ראשון נראה כי למוצר אין פעולות. ואכן, המוצר האמיתי הוא דומם, ואינו עושה כלום. עם זאת, העצם "מוצר" בתוכנית שלנו יידרש לבצע מספר פעולות, שמהותן היא אחזור ערכי תכונותיו.

מוצר
תכונות: מספר ברקוד: 0152439 מחיר: 15.00 מספר מדור: 3
שיטות: אחזר מחיר() אחזר מספר ברקוד() אחזר מספר מדור()
איור 1.7 – העצם "מוצר" על כל איבריו

הדוגמה שהבאנו בסעיף זה מציגה רק חלק קטן מתהליך הפיתוח של תוכנית מונחית עצמים. בתוכנית מלאה סביר שנמדל עצמים רבים ושונים, ונשקיע מחשבה רבה בחלוקת התפקידים והקשרים ביניהם. בנוסף, ברור שכדי להשלים את פיתוח העצמים "מוצר" ו"קופה" נצטרך גם לממש את השיטות ואת התכונות, אך נושא זה אינו מענייננו בפרק זה. אולי הבחנתם כי השיטות שתיארנו פשוטות יותר מאלו המתרחשות בסופרמרקט אמיתי. במערכת תוכנה אמיתית נזדקק לעצמים שתכונותיהם ושיטותיהם מורכבות יותר. לימוד יחידה זו יקנה חלק מהמיומנויות הדרושות למשימה כזו.

ה. תקשורת בין עצמים

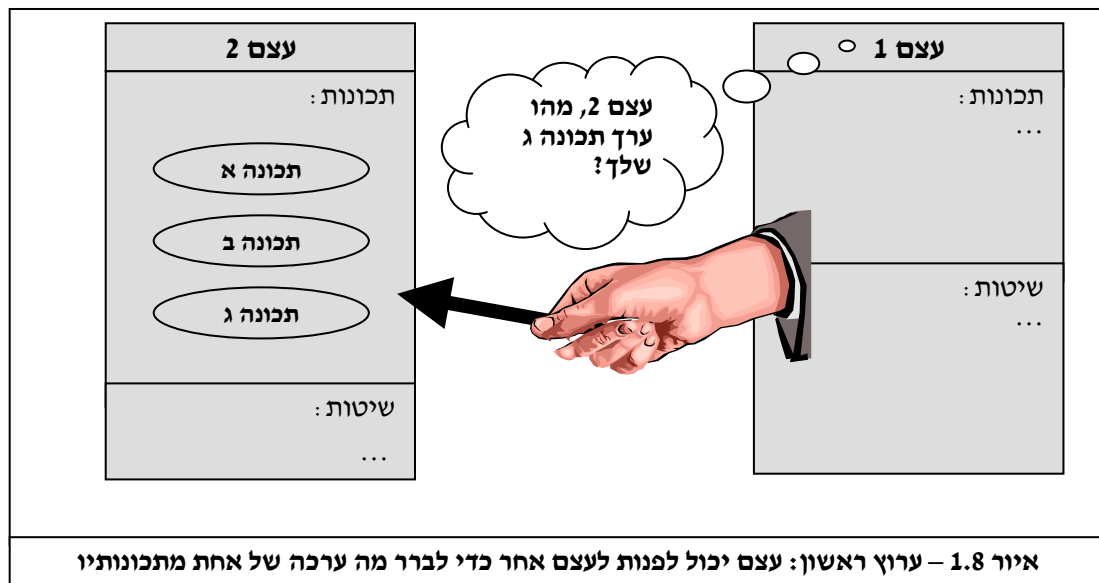
לא טוב היות העצם לבדו

בסעיפים הקודמים ערכנו היכרות ראשונית עם איברי העצם – השיטות והתכונות. אולם, העצמים בתוכנית אינם מביאים תועלת רבה כאשר כל אחד מהם פועל בנפרד. על מנת להתמודד עם משימות מורכבות, התוכנית צריכה לכלול מספר עצמים המשתמשים זה בזה ומתקשרים ביניהם. לדוגמה, בתוכנה המייצגת סופרמרקט, סביר להניח כי כדי לברר מהם מחירי המוצרים ולהנפיק חשבונות ה"קופות" יתקשרו עם ה"מוצרים". אם נרצה שהתוכנה תברר גם באילו מדפים חסרים מוצרים והאם יש מוצרים במלאי על מנת למלא אותם בהתאם, נרצה גם שה"קופות" יעדכנו את "מחסני המלאי" בדבר הקנייה, ו"מחסני המלאי" יתקשרו עם ה"מדפים" כדי לעדכן העברה של מלאי חדש.

עצם בתוכנית יכול לתקשר עם עצם אחר על ידי פנייה לאחד מאיבריו: שיטה או תכונה.

ערוץ תקשורת ראשון: פנייה לתכונה של עצם

לכל תכונה, כפי שראינו, יש ערך המוצב בה. אנו יכולים לפנות לעצם כדי לברר מה ערכה של תכונה כלשהי שלו, או כדי להציב בה ערך חדש. הדבר מומחש באיור הבא:

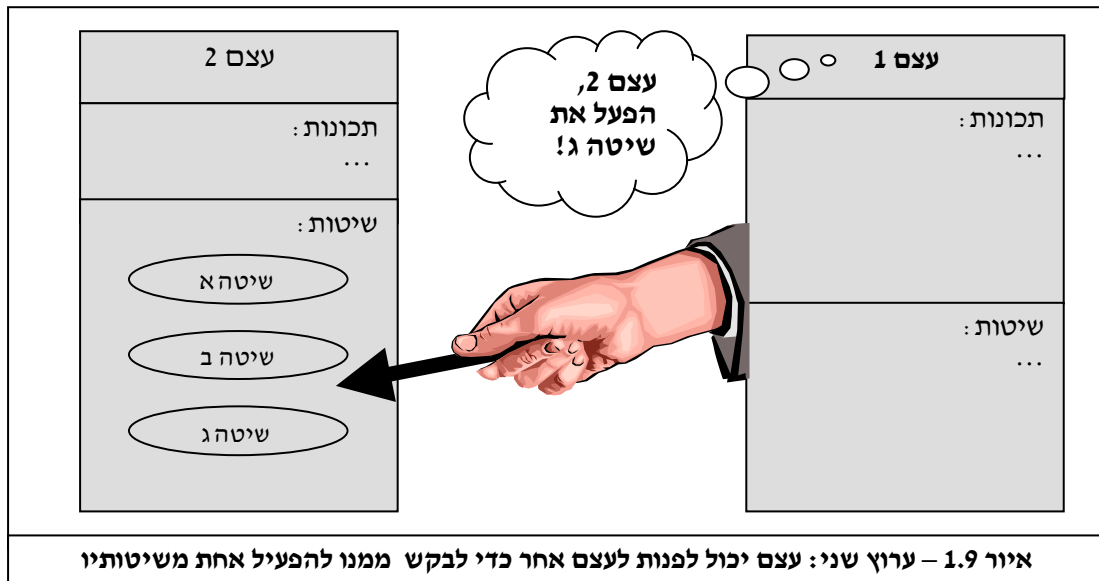


על מנת להדגים זאת ניזכר בשלוש השיטות של העצם "קופה" שפורטו בסעיף הקודם: שלם(מוצר), ברר-מחיר(מוצר) והזדכה(מוצר). כל השיטות הללו מקבלות מוצר, ואמורות לבצע פעולה כלשהי הקשורה בו. כדי לממש כל אחת מהשיטות הקופה תצטרך בין השאר לברר מה מחיר המוצר. לצורך כך הקופה תפנה למוצר ותקבל מידע על אודות ערכה של תכונתו "מחיר המוצר".

ערוץ תקשורת שני: פנייה לשיטה של עצם

בדרך כלל נעדיף לפנות לעצם אחר באמצעות שיטותיו. שיטות, כפי שהוסבר למעלה, הן פעולות שהעצם יודע לבצע, או במילים אחרות, שירותים שהעצם מוכן לספק לפונים אליו. כשאנו פונים לשיטה של עצם, אנו מבקשים מהעצם לבצע פעולה מסוימת או לספק לנו שירות מסוים. הפעולה או השירות המבוקשים יכולים להיות שינוי או אחזור של ערכי תכונות, ביצוע חישוב מתמטי וכן הלאה.

האיור הבא מדגים "פנייה לשיטה" המכונה גם "קריאה לשיטה" או "זימון שיטה".



ראינו כי הקופה יכולה לברר את מחיר המוצר על ידי פנייה ישירה לתכונותיו. דרך נוספת לעשות כן היא להפעיל את השיטה "אחזר-מחיר)". "שיטה זו מחזירה למי שפנה אליה את מחיר המוצר. הפעלת שיטה עדיפה בדרך כלל על פנייה ישירה לתכונה משום שהיא מממשת את עקרון ההכמסה (אנקפסולציה). עיקרון זה גורס שתכונות הן חלק מהמימוש הפנימי של העצם ולכן אינן צריכות להיות חשופות לשימוש על ידי גורמים חיצוניים. השיטות מהוות את ממשק העצם, וכל בקשת מידע או שינוי של מצב העצם צריכים (על פי רוב) להתבצע על ידי פנייה לשיטות.

1. המחלקה

מראה מעל הגשר

נעמוד על הגשר מעל הכביש המהיר ונשקיף מטה. מתחתינו חולפות מכוניות רבות ושונות: מרצדס, פיאט, טויוטה, פגיו. שוב פיאט. לכל רכב מספר זיהוי משלו, צבע משלו, מהירות משלו. עם זאת, ברור לנו כי כל העצמים החולפים תחת רגלינו הם מאותו הטיפוס, או במילים אחרות, כולם נמנים על אותה קבוצה: קבוצת המכוניות.

בסעיף זה נחدد את ההבחנה בין המכוניות החולפות בכביש לבין "מכונית" כללית, או בין המוצרים המונחים על המדף לבין "מוצר" כללי. במילים אחרות, נבדיל בין עצמים לבין הטיפוס שלהם. לשם כך נכיר שני מושגים חדשים: מחלקה ומופע.

מחלקה ומופע

מחלקה (class) – תבנית שלפיה ניתן ליצור עצמים מסוג מסוים או מטיפוס מסוים (טיפוס המחלקה). המחלקה אינה מייצגת עצם מסוים אלא הגדרה כללית של סוג של עצם, כגון: מכונית, מוצר, קופה וכדומה. סוג העצם קובע את התנהגותו, ולכן המחלקה כוללת רשימה של שיטות המשותפות לכל העצמים שיווצרו לפי תבנית המחלקה. כמו כן, סוג העצם קובע מה יהיו תכונותיו, ולכן כוללת המחלקה גם את רשימת התכונות של העצם, ללא פירוט הערכים שלהן. מחלקה מאפשרת, אם כן, ליצור עצמים מהטיפוס שאותו היא מייצגת.

מופע של מחלקה (instance) – עצם שנוצר לפי התבנית של מחלקה כלשהי, או במילים אחרות, עצם מן הסוג של המחלקה המסוימת. לכל המופעים של מחלקה כלשהי יהיו אותם איברים המפורטים בהגדרת המחלקה. להבדיל מהמחלקה שהיא תבנית מופשטת של עצם, למן הרגע שנוצר מופע יש בידינו עצם ממשי. לעצם כזה מוגדר מצב שאותו מייצגים הערכים המוצבים בתכונותיו.

? האם בין הערכים שמקבלות תכונות בשני מופעים שונים תמיד יהיה הבדל?

המחלקה כ"תבנית של עצמים"

קיימת הבחנה ברורה בין המחלקה, אשר מהווה הגדרה בלבד, לבין המופעים שלה, שנוצקו על פי ההגדרה. כדי להדגיש זאת, ניתן לדמות את מושג המחלקה לתבנית של עוגיות. לתבנית יש צורה וגודל קבועים, ולכן התבנית מגדירה את האפיון הבסיסי של העוגייה, ממש כפי שהמחלקה, שלה רשימת איברים קבועה, מגדירה את האפיון הבסיסי של כל מופע ששייך אליה. כל עוד לא יצקו בתבנית חומר לא יוצרו עוגיות. רק לאחר שניצוק בצק לתוך תבנית העוגיות (וכמובן נאפה אותן בתנור), נקבל עוגיות של ממש. לעוגיות אלו הגודל והצורה שהגדירה התבנית, אולם כל אחת מהן היא מופע בפני עצמו.

באופן דומה, כל עוד לא יצרנו מופעים מהמחלקה, בתוכנית אין עצמים מסוג מחלקה זו. רק לאחר שנייצר מופעים ממחלקה, יהיו בתוכנית זו עצמים של ממש. עצמים אלו יהיו מטיפוס המחלקה, ויתאימו להגדרתה. עם זאת, חשוב לציין כי כל מופע יהיה עצם נפרד. מספר המופעים שניצור מכל מחלקה תלוי במשימות התכנות שאותן אנו מנסים לפתור. לפיכך, אם למשל בסופרמרקט שאנו מייצגים יש חמש קופות, ניצור בתוכניתנו חמישה מופעים של המחלקה "קופה".

הקופה הראשית	המחלקה קופה	הקופה המשנית
תכונות : מספר סידורי : 1 סכום כסף : ₪ 45,000	תכונות : מספר סידורי סכום כסף	תכונות : מספר סידורי : 2 סכום כסף : ₪ 35,000
הקופה המהירה	שיטות : שלם(מוצר) ברר מחיר(מוצר) הזדכה(מוצר)	הקופה של אבי
תכונות : מספר סידורי : 3 סכום כסף : ₪ 12	הקופה של שולה	תכונות : מספר סידורי : 5 סכום כסף : ₪ 22,000
תכונות : מספר סידורי : 4 סכום כסף : ₪ 24,000		
איור 1.10 – המחלקה 'קופה' ומספר המופעים שלה. לכל המופעים אותן שיטות כמו אלו המוגדרות עבור המחלקה, ולכן איננו מציינים אותן מחדש בכל מופע		

שיטה-בונה

ראינו שמופעים מיוצרים לפי התבנית שמגדירה המחלקה. אולם כיצד יוצרים עצמים מהמחלקה? בכל מחלקה קיימת שיטה מיוחדת הנקראת "שיטה-בונה" (constructor) המאפשרת ליצור מופעים לפי תבנית המחלקה. שיטה זו יוצרת מופע חדש מסוג המחלקה, ולרוב גם מאתחלת את תכונותיו, כלומר מציבה בהן ערכים. ערכים אלו קובעים את מצבו של המופע החדש שנוצר.

כדי להבין מהי שיטה-בונה, ניתן לדמות את המחלקה לפס ייצור. במפעל מכוניות, למשל, קיים פס ייצור המסוגל לייצר מכוניות רבות, כולן מאותו הדגם. אמנם כל המכוניות המיוצרות הן בעלות תכונות משותפות רבות, אך לכל אחת יש גם מאפיינים ייחודיים. למשל, ניתן ליצור מכוניות בצבעים שונים. פס הייצור מגדיר באופן כללי את תבנית המכונית שממנה ניתן ליצור מכוניות שונות. השיטה-הבונה, כמוה ככפתור המפעיל את פס הייצור בכדי ליצור מכונית נוספת.

המחלקה קופה
תכונות : מספר סידורי סכום כסף
שיטות : שיטה-בונה לקופה()
שלם(מוצר) ברר מחיר(מוצר) הזדכה(מוצר)
איור 1.11 – המחלקה קופה לרבות שיטה-בונה

ז. ירושה

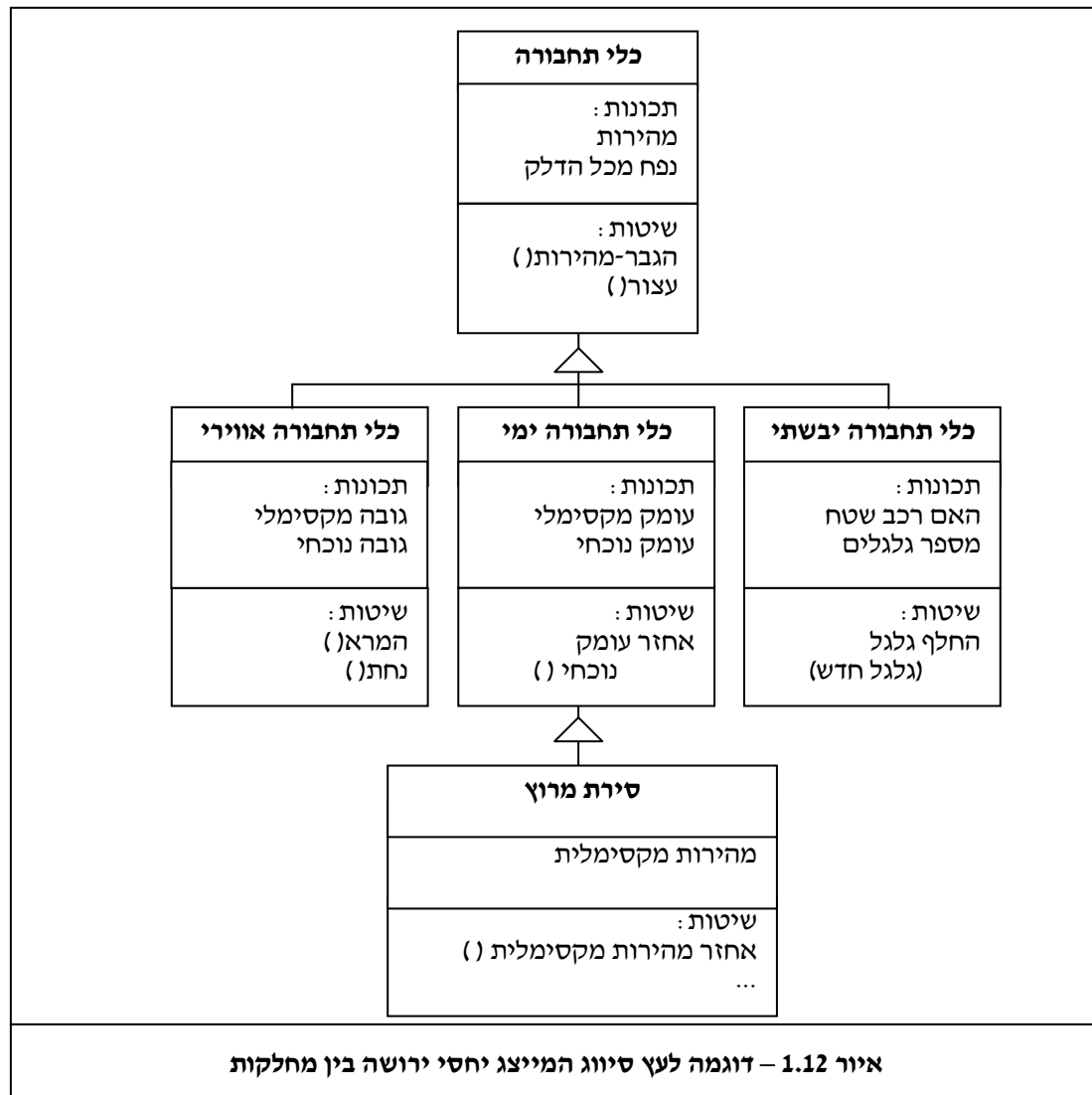
ראינו כי כל תוכנית מונחית עצמים מורכבת ממחלקות, שמגדירות סוגים של עצמים, ומעצמים שנוצרו על פי מחלקות אלו. גישת ה-OOP תומכת לא רק בייצוג עצמים ומחלקות, אלא גם בייצוג יחסים בין מחלקות.

יחסי סיווג ומנגנון הירושה

היחס החשוב ביותר שנוכל לייצג בין מחלקות בתכנות מונחה עצמים הוא יחס סיווג. סיווגים רבים מוכרים לנו מחיי היום-יום: כלב הוא סוג של יונק ויונק הוא סוג של בעל חיים. משולש הוא סוג של צורה גיאומטרית, ומכונית היא סוג של כלי רכב. פעמים רבות נרצה לייצג יחסים אלו גם בתוכנית שלנו. נניח למשל כי הגדרנו את המחלקה "מכונית" אך גם את המחלקה "כלי רכב". ברור כי יש קשר הדוק בין שתי המחלקות. אם למחלקה "כלי רכב" יש תכונות כגון "מהירות", ושיטות כגון "התנע" ("או "הגבר מהירות"), אנו רוצים כי גם למכונית תהיינה שיטות ותכונות אלו. אנו מצפים שאם נייצג בתוכנית את הקשר בין המחלקה "מכונית" והמחלקה "כלי רכב" נקבל ב"מכונית" את כל התכונות והשיטות של "כלי רכב", בלי שנצטרך להגדיר אותן במפורש.

שפות תכנות מונחה עצמים מאפשרות לנו לייצג יחס זה בין מחלקות באמצעות מנגנון המכונה **ירושה (inheritance)**. באמצעות שימוש בירושה ניתן להגדיר מבנה היררכי בין מחלקות שונות: נוכל להגדיר למשל כי המחלקה "מכונית" **יורשת** מן המחלקה "כלי רכב". המחלקה המקורית (במקרה זה "כלי רכב") תקרא **מחלקת-על (super class)**. המחלקה היורשת (במקרה זה "מכונית"), תקרא **תת-מחלקה (subclass)**. תת-המחלקה תהיה תמיד "**סוג של**" מחלקת-העל, ותקבל "בירושה" באופן אוטומטי את כל האיברים (שיטות ותכונות) של מחלקת-העל.

לצורך הדוגמה התבוננו באיור הבא. איור זה הוא דוגמה לעץ סיווג המייצג יחסים בין מחלקות שונות. בפרק "ירושה ופולימורפיזם" תלמדו בצורה מסודרת כיצד לנתח ולכתוב עצים כאלו. לעת עתה שימו לב כי בתת-המחלקות לא ציינו את האיברים של מחלקות-העל, שכן אלו מתקבלים באופן אוטומטי, אם הגדרנו את יחסי הירושה בין המחלקות.



? איזו שיטה חסרה בכל המחלקות המופיעות באיור 1.12?

קשה להבין את היתרונות הטמונים במנגנון הירושה בלי ללמוד אותו לעומקו, ובכל זאת נציין כי מנגנון הירושה מאפשר לנו לייצג את היחסים בין מחלקות שונות בצורה מדויקת וליצור בתוכניתנו מידול טוב יותר של ה"עולם". העובדה שתת-המחלקות יורשות באופן אוטומטי את כל האיברים של מחלקת-העל היא מימוש של עקרון השימוש החוזר בקוד. מנגנון הירושה יאפשר לנו גם להסתכל על עצמים בצורה רב-גונית: לעתים נוכל לראות במכונית בימבה עצם מהטיפוס "מכונית", ולעתים עצם מהטיפוס "כלי רכב".

רעיון הירושה הוא מעמודי התווך של תכנות מונחה עצמים. בפרק הדין בירושה נכיר את המנגנון הזה ואת יתרונותיו באופן מעמיק יותר, ונבין מדוע הוא כלי תכנות רב עוצמה.

ח. סיכום

בפרק זה התוודענו לראשונה אל **תכנות מונחה עצמים**. למדנו להתבונן בעולם דרך משקפת מיוחדת: משקפת OOP, שדרכה המציאות נראית כאוסף של עצמים שלהם **תכונות ושיטות**. מאוחר יותר, הבנו כי כל עצם הוא למעשה **מופע של מחלקה** מסוימת. הכרנו שיטה מיוחדת הקיימת בכל מחלקה ומאפשרת ליצור עצמים חדשים: **השיטה-הבונה**.

ראינו כי תכנות מונחה עצמים מיישם עקרונות תכנותיים שהכרנו בעבר, כמו מודולריות וכמו שימוש חוזר בקוד (code reuse). לבסוף, פגשנו לראשונה את אחד העקרונות המרכזיים בתכנות מונחה עצמים, עיקרון ה**ירושה**, המאפשר לסדר מחלקות בהיררכיה, באופן שבו כל **תת-מחלקה** יורשת מ**מחלקת-העל** שלה את כל איבריה.

לתכנות מונחה עצמים יתרונות רבים. גישה זו מאפשרת לבנות תוכניות שלהן מבנה פשוט ואינטואיטיבי. היכולת לחשוב גם על מערכות גדולות באופן אינטואיטיבי היא יתרון הן בתהליך הפיתוח של תוכנית, הן ביכולת לקרוא את הקוד ולתקנו בעתיד.

כפי שראינו, בתכנות מונחה עצמים מוצג כל עצם כ"חבילת קוד" נפרדת ועצמאית, תוך שאיפה לצמצם במידת האפשר את התלות בין "חבילות הקוד" השונות. היכולת לחלק את הקוד לחלקים עצמאיים מכונה **מודולריות**. היכולת "לעטוף" כל אחד מהחלקים ביחידת קוד סגורה שלה פעולות המאפשרות תקשורת עם החוץ מכונה **אנקפסולציה** (או **הכמסה**).

שתי יכולות אלו מקנות לתכנות מונחה עצמים יתרונות נוספים. קל יחסית לבדוק, לתקן או לשנות את הקוד שכן ניתן לשנות עצמים ומחלקות מסוימים בלי לפגוע ביתר התוכנית. בנוסף, ניתן לשלב עצמים ומחלקות ממקורות שונים: למשל, ניתן לשלב בתוכנית חדשה מחלקות מתוכנית אחרת, ולייצר מהן עצמים שיסייעו במימוש מטרות התוכנית.

ניתן לממש תכנות מונחה עצמים בשפות רבות. אנו נלמד את שפת ג'אווה, ש"נולדה וגדלה" בתוך עולם ה-OOP, ולכן נוחה ביותר לתכנות מסוג זה. בפרקים הבאים נכיר את יסודותיה של שפת ג'אווה, ונבין כיצד יסודות אלו מותאמים לתכנות OOP.

ט. לפני שמתחילים...

שתי הערות חשובות לפני שאנו צוללים אל מעמקי הספר:

- במקרים מסוימים יהיה חלקכם משוכנע כי דווקא גישת התכנות המוכרת לכם מלימודיכם הקודמים טובה, מהירה ויעילה יותר מגישת תכנות מונחה עצמים. לא אחת יהיה הצדק עם המעדיפים את שפות התכנות המוכרות לכם. למשל, כדי לכתוב תוכנית המדפיסה את המשפט "hello world" על המסך, תוכלו לכתוב תוכנית קצרה ופשוטה ביותר ב-C או בפסקל. בשפה מונחית עצמים, לעומת זאת, ייתכן מאוד שתצטרכו להגדיר לשם כך תוכנית מסורבלת יותר.

תכנות מונחה עצמים הוא כלי מקובל המאפשר ליצור תוכניות ברורות ואינטואיטיביות בזמן קצר יחסית. יתרונותיה של שיטת תכנות זו באות לידי ביטוי ככל שמשימת התכנות הניצבת בפנינו מורכבת יותר.

- ההחלטה כיצד נמדל את העולם האמיתי, או כיצד נמדל משימה שאתה אנו צריכים להתמודד אינה פשוטה כלל. עלינו להגדיר אילו אובייקטים נייצג בתוכניתנו על ידי עצמים, ומאילו נתעלם. הדבר דורש מחשבה וניסיון רב. לדוגמה, יהיה מי שיחליט לכלול בתוכנה לסופרמרקט דווקא את העצמים: "סניף", "לקוח" ו"עסקה". אחר יעדיף להוסיף את העצם "רשימת-מוצרים" שיאגור את מלאי המוצרים ומחיריהם. חשוב להבין שבמקרה זה, כמו בכל תכנון של מערכות תוכנה, אין פתרון יחיד לבעיה. ייתכנו כמה פתרונות טובים, שלכל אחד יתרונות משלו.

מושגים

member	איבר
encapsulation	הכמסה/אנקפסולציה
inheritance	ירושה
modularity	מודולריות
instance	מופע של מחלקה
class	מחלקה
super class	מחלקת-על
interface	ממשק
design	עיצוב
object	עצם
method	שיטה
constructor	שיטה-בונה
code reuse	שימוש חוזר בקוד
attribute	תכונה
Object Oriented Programming (OOP)	תכנות מונחה עצמים (תמ"ע)
subclass	תת-מחלקה

פרק 1 דף עבודה מס' 1

מידול עצמים בעזרת נייר ועיפרון

מטרות

התנסות בהגדרת מחלקות (על פי עצמים קיימים).

מה עליכם לעשות?

לפניכם רשימת עצמים :

- דלת
- דלת אוטומטית
- מאוורר
- מכונה אוטומטית לממכר פחיות
- מצלמת וידאו
- שעון יד דיגיטלי
- לווין צילום

ברצוננו להגדיר מחלקה בעבור כל עצם ברשימה. לשם כך עליכם להיעזר בתרשימים המופיעים
בנספח 1.
בעבור כל מחלקה, ציינו תכונות אחדות וכמה שיטות עיקריות המאפיינות אותה. זכרו לכלול
שיטה-בונה בכל מחלקה.
בעבור כל שיטה ציינו אם היא מקבלת ערכים או מחזירה ערכים.

בהצלחה!

פרק 1 דף עבודה מס' 2

מידול עצמים – ירושה

מטרות

הגדרת מחלקות שיש ביניהן יחס ירושה.

מה עליכם לעשות?

לפניכם רשימת עצמים :

- מאורר
- רדיו
- טלוויזיה
- מצנע
- מזגן אוויר
- מחשב ביתי
- מיקרוגל
- מכשיר חשמלי
- שואב אבק
- מכשיר לחימום מזון

ברצוננו להגדיר את המחלקות שמהן נוצרו העצמים שברשימה ואת יחסי הירושה המתקיימים ביניהן. לשם כך עליכם לשרטט עץ סיווג מתאים. בעבור כל מחלקה שאתם ממקמים בעץ ציינו תכונות אחדות ומספר שיטות. זכרו לכלול שיטה-בונה בכל מחלקה. בתום בניית העץ וודאו כי איברי מחלקת-על אינם מופיעים שנית בתת-המחלקות שלה (שכן אלו ממילא יורשות את כל האיברים הללו, ואין לצייןם שוב). השתמשו בצורת האיור המקובלת לסימון מחלקות. דוגמאות תוכלו למצוא בנספח 1. אם מחלקה יורשת ממחלקה אחרת ציינו זאת על ידי שימוש באיור המייצג יחס ירושה (ראו נספח).

בהצלחה!