

פרק 6

על עצמים וטיפוסים

צ'מ'ס וט'כוס'ס זלט' מה?

בתכנות מונחה עצמים, תהליך פתרון בעיה מתחיל בזיהוי המרכיבים המשתתפים בפתרון. למרכיבים אלו אנו קוראים עצמים. עצמים ניתן לסווג לסוגים שונים. עבור כל סוג של עצמים מגדירים טיפוס הכולל את המאפיינים המשותפים לכל העצמים מאותו סוג. אלגוריתם לפתרון בעיה כולל גם הוראות הבונות עצמים על פי תבנית הטיפוסים, והוראות המפעילות פעולות על העצמים האלו.

בפרק נלמד:

- ✓ חלק א: עקרונות תכנות מונחה עצמים - עמוד 116
 - מהו עצם?
 - תכונות ופעולות
 - תכונות וערכי תכונות
 - מהו טיפוס?
- ✓ חלק ב: תכנות עם עצמים - עמוד 123
 - בניית עצמים
 - הפעלת פעולות על עצמים
- ✓ חלק ג: פיתוח מחלקה המייצגת טיפוס - עמוד 129
 - מבנה מחלקה המייצגת טיפוס
- ✓ חלק ד: פיתוח מחלקה ראשית המשתמשת בעצמים - עמוד 135



.....

.....

.....

.....

חלק א: עקרונות תכנות מונחה עצמים

מהו צמ?

אם נסתכל סביב נוכל לראות דוגמאות רבות של עצמים מהעולם האמיתי: שולחן הכתיבה שלך, העט שלך, המורה של הכיתה, המחשב של התלמיד שיושב לידכם, השעון שלו, הכלב של השכן ועוד ועוד. למעשה עצמים הם המרכיבים הבסיסיים של העולם. עצמים אינם רק חפצים ובעלי חיים. גם האהבה של יוסי לחנה היא עצם, הריקוד שרקדתם אתמול בדיסקוטק, שיחת הטלפון שניהלת שלשום עם חיים מיוד 4 ועוד ועוד. עצם הוא כל דבר... כמעט. הבנת המושג עצם היא המפתח להבנת תכנות מונחה עצמים.

תכונות וכפולות

את כל העצמים ניתן לתאר על ידי שני מאפיינים: לכל עצם יש **תכונות** המתארות אותו ו**פעולות** שניתן לבצע עליו. לתכונות יש ערכים המאפיינים עצם מסוים – **ערכי תכונות**. הפעולות הן פעולות אפשריות לביצוע, ולא פעולות המתבצעות מעצמן. אנו יכולים להפעיל על עצם את כל אחת מהפעולות כמה פעמים, או אפילו לא פעם אחת. הפעלת פעולה היא רק לפי בקשה. מכלול התכונות וערכי התכונות מייצגים יחד את **מצב העצם**. בכל רגע נתון יש לעצם את מצב העצם. הפעלת פעולה אשר משנה את הערך של אחת מן התכונות של העצם, משנה למעשה את מצב העצם. פעולות שאינן מתייחסות אל תכונות העצם הן חסרות משמעות ולכן לא יכללו באוסף הפעולות של העצם.

ניקח לדוגמה את **העצם - הטלפון הסלולרי שלך**.

תכונותיו הן: מספרו, שם הרשת לה הוא מחובר, שם בעליו, אחוז הטעינה של הסוללה, המספרים שנמצאים בזיכרון.

הערכים של התכונות בהתאמה הם למשל: 054-5533447, אורנג', אהרון הכהן, 65%, אין מספרים בזכרון.

ניתן לבצע עליו פעולות לדוגמה: להתקשר ממנו, למכור אותו, להוסיף מספר לזיכרון, למחוק מספר מהזיכרון, לטעון אותו.

ביצוע פעולות יכול לשנות את ערכי התכונות של העצם – הטלפון הסלולרי שלך: מכירת הטלפון תשנה את הבעלים, מחיקת מספר מהזיכרון תשנה את רשימת המספרים המצויים בזיכרון, טעינת הטלפון תשנה את אחוז הטעינה של הסוללה.

בקבוק הבושם שלך הוא דוגמה נוספת לעצם.

התכונות הן: שם הבושם שהוא מכיל, מחיר הקניה, כמות הבושם שנותרה בבקבוק, היכן הוא נמצא. **ערכי התכונות בהתאמה הם** למשל: יסמין, 250 ש"ח, 35 מ"ל, ארון חדר האמבט של יעל.

ניתן לבצע עליו את הפעולות: לרסס ממנו כמות מסוימת, לשים אותו בחדר השינה, לבדוק מה כמות הבושם שנותרה, לבדוק האם בבקבוק הבושם ריק?

גם ביצוע פעולות אלו עשוי לשנות את ערכי התכונות של בקבוק הבושם, למשל, ריסוס בושם יקטין את כמות הבושם בבקבוק.



ייצוג של עצם בתרשים

נשתמש בתרשים כדי להציג עצם. תרשים של עצם מורכב משלושה חלקים: שם העצם, רשימת התכונות וערכיהן, והפעולות שניתן להפעיל על העצם. כאשר פעולה מקבלת פרמטרים הם יופיעו בתוך סוגריים, לפני שם הפרמטר יופיע קו תחתון (_) המסמן שזה הוא פרמטר.

דוגמה לעצם: העיפרון המגניב של רוני

העצם	העיפרון המגניב של רוני
תכונות וערכיהן	צבע: צהוב עובי העופרת: 0.5 אורך: 10 ס"מ מחיר: 3 ₪ מיקום: על השולחן
פעולות	מה אורך העיפרון? חדד את העיפרון הכנס העיפרון לקלמר האם העיפרון נמצא בקלמר?



התרשים מציג עצם ששמו הוא העיפרון המגניב של רוני, עם התכונות צבע (ערך: צהוב), עובי העופרת (ערך: 0.5), אורך (ערך: 10 ס"מ), מחיר (ערך: 3 ש"ח) ומיקום (ערך: על השולחן). בחלק התחתון של התרשים מופיעה רשימת הפעולות שניתן לבצע על העיפרון: מה האורך?, חדד, הכנס לקלמר ו האם העיפרון נמצא בקלמר?.

דוגמה לעצם: פחית המשקה של ניצה

העצם	פחית המשקה של ניצה
תכונות וערכיהן	משקה: דיאט קולה נפח הפחית: 330 מ"ל כמות המשקה: 100 מ"ל מחיר: 6 ₪ האם הפחית פתוחה: לא
פעולות	פתח שתה (_ כמות) מה המחיר?



לעצם פחית המשקה של ניצה יש מספר תכונות: המשקה שהיא מכילה (דיאט קולה), נפח הפחית (330 מ"ל), כמות המשקה שבפחית (100 מ"ל), מחיר הקניה (6 ₪), האם הפחית פתוחה (לא). הפעולות שניתן לבצע על הפחית הן: פתח את הפחית, שתה כמות מסוימת מהמשקה, בדוק מהו מחיר הקניה.

נבדוק את הקשרים בין הפעולות לבין התכונות בעצם פחית המשקה של ניצה:

- כאשר הפחית סגורה, פעולת פתח גורמת לשינוי ערך התכונה האם הפחית פתוחה?, אחרי שהפעולה תתבצע ערך התכונה האם הפחית פתוחה יהיה "כן". שים לב: לא חשוב מה היה ערך התכונה קודם, בכל מקרה ערך התכונה החדש יהיה "כן".
- פעולת מה המחיר? מאחזרת את ערך התכונה מחיר. כלומר, הפעולה תחזיר את הערך הנוכחי של התכונה מחיר. פעולה זו אינה משנה את הערך של תכונה כלשהי.
- כדי לבצע את הפעולה שתה, חייבים לצרף מידע נוסף: מה כמות השתיה שאותה שותים. בכל הפעלה של הפעולה, כמות השתייה יכולה להיות שונה ולכן מופיע בה הפרמטר _כמות המייצג את כמות המשקה. ערך זה יכול להשתנות מהפעלה אחת של הפעולה להפעלה אחרת. למשל: שתה(50) היא הפעלה של הפעולה לשתית 50 מ"ל מהמשקה. פעולות חייבות להיות מוגדרות היטב לכל מצב אפשרי של העצם. למשל יש להחליט מה קורה אם בפחית יש 30 מ"ל ויש הפעלה של הפעולה שתה(50). החלטה אפשרית היא למשל: אם הכמות המבוקשת גדולה מהכמות בפחית, הפחית תתרוקן. החלטה אחרת אפשרית היא לתת בנוסף הודעה שחסרים 20 מ"ל.

- אם היינו מגדירים פעולה נוספת צבע את הפחית(לצבע), שמטרתה לשנות את צבע הפחית, והיינו מפעילים אותה למשל כך: צבע את הפחית(אדום), האם יש לפעולה משמעות? לפעולה זו אין משמעות מאחר ואין לנו תכונה של צבע הפחית, מצב העצם יהיה זהה לפני ואחרי הפעלת הפעולה, כי הפעלתה לא יכולה להיות ניכרת בו ולכן יש להימנע מהגדרת פעולות מסוג זה.

דוגמה לעצם: השיר הללויה

ל"שיר הללויה" יש מספר תכונות: שם המשורר, שם המלחין, מילות השיר, לחן השיר. הפעולות שניתן לבצע על השיר הן: מי המלחין?, השמע ברדיו, שנה סולם (לכלי נגינה).



השיר הללויה	העצם
שם המשורר: שמרית אור שם המלחין: קובי אושרת מילות השיר: "הללויה לעולם, הללויה ישירו כולם..." לחן השיר: "G D G D Bm..."	תכונות וערכיהן
מי המלחין? השמע ברדיו שנה סולם (לכלי נגינה)	פעולות

שים ♥ : עצם אינו רק "עצם" מוחשי. ניתן להגדיר עצמים גם עבור דברים שהם מופשטים, כל דבר יכול להיות עצם אם יש לו תכונות ו אם אפשר להגדיר פעולות המתייחסות לתכונות. השיר קיים בין אם הוא רשום על דפי תווים או לא, בין אם הוא צרוב על דיסק בין לא, בין אם הוא מושמע או לא. עצם יכול להיות למשל החלום שחלמתי הלילה, או רעיון מסוים שהגה המדען חכמוני לפני חודש.

כתוב תרשים תרג'לים באכ"ן 38 נ"מ

שים ♥ : בהגדרת הפעולות התייחס אל פעולות המקבלות פרמטרים.

תרגיל 1: הכבשה דולי ★



הצג בעזרת תרשים את העצם: הכבשה דולי מקיבוץ עין חרוד. לדולי יש את התכונות: שם, שם המטפל, כמות הצמר עליה, כמות החלב שיש לה. ניתן לבצע עליה פעולות: לאחזר מי המטפל, למכור אותה למישהו, לחלוב אותה ולגזוז אותה.

תרגיל 2: המורה שמוליק ★



הצג בעזרת תרשים את העצם: המורה שמוליק דוד מביה"ס טל. לשמוליק יש את התכונות: שם, מין, שכר, רשימת הכיתות בהן הוא מלמד, דרגה. ניתן לבצע עליו את פעולות: לאחזר מה השכר שלו, להעלות את שכרו, לשבץ אותו להוראה בכיתה נוספת, לשאול האם הוא רעב, לתת לו לאכול מאכל. האם כל הפעולות שהוגדרו מתאימות?

תרגיל 3: העצם הראשון שלי ★



בחר איזה עצם ברצונך לייצג והצג את המידע עליו (תכונות, ערכי תכונות, פעולות) בעזרת תרשים של עצם.
(ב) תכונות וערכי תכונות

תכונות וארכי תכונות

- לכל עצם יש תכונות בעלות ערכים. מכלול התכונות והערכים שלהן מהווים את **מצב העצם**. ביחס לתכונות חשוב להדגיש את הדברים הבאים:
 - בכל מצב (ברגע נתון) לכל תכונה יש ערך אחד.
 - לדוגמה: עבור העצם **פחית המשקה של ניצה**, לתכונה **כמות המשקה** יש את הערך 100.
 - ניתן לשנות ערך של תכונה על ידי הפעלת פעולה מתאימה.
 - לדוגמה: על העצם **פחית המשקה של ניצה**, שערך התכונה **כמות המשקה** שלה היא 100, אפשר להפעיל את הפעולה **שתה(40)**, ובעקבותיה יהיה ערך התכונה **כמות המשקה** 60.
 - ערכי התכונות בעצמים שונים מאותו טיפוס יכולים להיות זהים או שונים.
 - לדוגמה: ערך התכונה **משורר** יהיה **אהוד מנור** במספר עצמים שהם מטיפוס שיר והערך יהיה **עזי חיטמן** בעצמים אחרים.
 - כל עצם נבנה באופן מפורש ויש לו מזהה. העצמים נפרדים זה מזה גם אם ערכי כל התכונות שלהם זהים.
 - לדוגמה: מכוניות מסוג מסוים שיצאו מפס ייצור הן בעלות ערכי תכונות זהים בכל הקשור לנפח מנוע, אורך ורוחב המכונית ומספר המושבים. עדיין כל מכונית היא עצם בפני עצמו ויש לה מזהה.
 - לתכונה יש תמיד ערך. הערך הראשוני נקבע בעת בניית העצם. ערכי התכונות בשלב בניית העצם יכולים להקבע בדרכים שונות: השמת ערכים קבועים לתכונות, השמת ערכים לתכונות על פי ערכי פרמטרים, השמת ערכים לתכונות על פי קלט, או כאשר אין השמות מפורשות יקבעו ערכי ברירת מחדל לכל תכונה על פי הטיפוס שלה. דרך קביעת הערכים לא צריכה להיות זהה עבור כל התכונות. כלומר, תכונה אחת יכולה לקבל ערך קבוע, שנייה לקבל ערך על פי פרמטר ושלישית לקבל ערך של ברירת מחדל.
 - לדוגמה: העצם **שיחת טלפון בין רוני ליעל**. מאחר והשיחה מקומית המחיר לדקת שיחה יקבע באופן אוטומטי על ידי קבוע, זמן תחילת השיחה יקבע על פי פרמטרים, זמן סיום השיחה עדיין אינו ידוע לכן בבניית העצם יקבל ערך ברירת מחדל של 0, והוא יעודכן כאשר השיחה תסתיים.

טיפוס תכונה

- לכל תכונה מוגדר הטיפוס שלה. בשפת התכנות הכרנו סוגים שונים של טיפוסים עבור משתנים. הטיפוסים המתאימים למשתנים מתאימים גם לתכונות. לכל תכונה נבחר את הטיפוס המתאים לערכים שלה על פי סוגם. להלן מספר דוגמאות של סוגים של תכונות:

שם התכונה	ערך התכונה הוא מסוג	הטיפוס ב C#	דוגמאות
כמות	מספר שלם	int	200, 6, 3
אורך	מספר ממשי	double	53.111, 84.5, 130.00
שם	מספר מילים	string	"משה כהן", "לאה אור", "בת-אל בר"
תאריך	מספר מילים במבנה אחיד	string	"23 מרץ 1999", "6 מאי 2000"
רעב?	כן/לא	bool	false, true
טמפרטורות	מערך שלמים	int []	28, 25, 20, 19, 22

למעשה טיפוס של תכונה יכול להיות לא רק מטיפוסים הבנויים בשפה אלא גם מטיפוסים המוגדרים על ידי המשתמש, כמו אלה שאת הגדרתם אנו לומדים בפרק זה. טיפוס המשתמש בטיפוסים אחרים של המשתמש נקרא טיפוס מורכב ולא נכלל בספר זה.

מהו טיפוס?

בדיון על עצמים ראינו כי כל עצם מאופיין על ידי התכונות שלו (וערכיהן בהתאמה) ועל ידי הפעולות שניתן להפעיל עליו. בדרך כלל יש קבוצה של עצמים שלכולם יש את אותם מאפיינים, כלומר תכונות ופעולות זהות. כדי לנצל את המכנה המשותף, ולהמנע מאפיון מחדש עבור כל עצם, נגדיר טיפוס. הטיפוס יאפיין את התכונות והפעולות, ובהתאם להגדרתו נוכל לבנות עצמים מן הטיפוס. לכל העצמים שיבנו על פי טיפוס יש את כל התכונות המוגדרות בו ואת כל הפעולות המוגדרות בו. **טיפוס הוא תבנית המאפיינת עצמים**. בטיפוס נגדיר את התכונות שיהיו לכל עצם שיבנה על פי הטיפוס, וגם את הפעולות שניתן להפעיל על כל עצם שיבנה על פי הטיפוס. לכל עצם שיבנה יהיה מזהה - **מזהה העצם**. בעזרת המזהה נוכל להבדיל בין עצמים שונים מאותו טיפוס. לכל עצם יהיה את **מצב העצם** שלו, כלומר ערכי התכונות המאפיינות אותו. תמיד כשמתייחסים אל עצם יש לומר מאיזה טיפוס הוא.

דוגמה לטיפוס: מחברת

ניתן לאפיין תכונות ופעולות המשותפים לכל העצמים שהם מחברת. לכל מחברת בעולם יש את התכונות: **גודל דף, סוג דף, מספר דפים**, ועל כל המחברות בעולם ניתן לבצע את הפעולות: **תלוש דף**, כתוב על דף, האם יש דף ריק? אוסף תכונות ופעולות זה הוא הטיפוס מחברת. הטיפוס מחברת הוא תבנית לבניית עצמים מטיפוס זה. כלומר, מחברת שהיא עצם מסוים, תבנה על פי התבנית הזו. עצם מטיפוס מחברת יכול להיות למשל **מחברת ההיסטוריה של פלג**. עצם זה נבנה על פי תבנית הטיפוס מחברת, כלומר יש לו את כל התכונות המוגדרות בטיפוס מחברת, וגם ניתן להפעיל עליו את כל הפעולות המוגדרות בטיפוס מחברת. מאחר ו **מחברת ההיסטוריה של יעל** היא מחברת מסויימת יש לה את ערכי התכונות המתאימים לה: **גודל הדף שלה הוא A4, סוג הדף הוא 80 גר', ו מספר הדפים בה הוא 72**. לעצם **מחברת ההיסטוריה של גל**, יש את אותן התכונות עם הערכים המתאימים לה (שיכולים להיות זהים או שונים לאלו של מחברת ההיסטוריה של פלג).

דוגמה לטיפוס: מחשב

הבה נתבונן בשלושה עצמים:

העצם	מחשב1 של יפתח	העצם	מחשב2 של יפתח	העצם	המחשב של מעין
תכונות וערכיהן	בעלים: יפתח מור מערכת הפעלה: חלונות XP	תכונות וערכיהן	בעלים: יפתח מור מערכת הפעלה: חלונות Me	תכונות וערכיהן	בעלים: מעין טל מערכת הפעלה: חלונות XP
פעולות	זיכרון: Mg128 האם דולק: לא דיסק קשיח: 1G מיקום: חדר עבודה	פעולות	זיכרון: Mg256 האם דולק: כן דיסק קשיח: 1G מיקום: משרד	פעולות	זיכרון: Mg128 האם דולק: לא דיסק קשיח: 2G מיקום: החדר של מעין
פעולות	הדלק כבה הגדל זיכרון (_ כמה)	פעולות	הדלק כבה הגדל זיכרון (_ כמה)	פעולות	הדלק כבה הגדל זיכרון (_ כמה)



לכל העצמים תכונות משותפות: בעלים, מערכת הפעלה, זיכרון, האם דולק, דיסק קשיח ומיקום. ערכי חלק מהתכונות זהים ואחרים שונים, אולם ברור שיש כאן שלושה עצמים שונים. כמו כן, עבור כולם מוגדרות פעולות זהות: הדלק, כבה, הגדל זיכרון. הבסיס המשותף מאפשר להגדיר את הטיפוס "מחשב" כאוסף התכונות והפעולות המשותפות לכל העצמים האלה. הטיפוס מחשב יהיה תבנית לבניית עצמים מטיפוס מחשב. לכל עצם שיבנה מן הטיפוס תהיינה אותן תכונות, וניתן יהיה להפעיל עליו את אותן פעולות.

ייצוג של טיפוס בתרשים

נשתמש בתרשים כדי להציג טיפוס. תרשים של טיפוס מורכב משלושה חלקים: שם הטיפוס, רשימת התכונות המאפיינות עצמים מן הטיפוס, ורשימת פעולות שניתן להפעיל על עצמים מן הטיפוס. בתרשים העצם יחול שינוי מאחר ואין צורך לכלול בו את הפעולות שניתן לבצע עליו. הפעולות מוגדרות באופן כללי בטיפוס, כי ניתן להפעיל אותן על כל עצם מן הטיפוס. אם נרצה לדעת אילו פעולות ניתן להפעיל על עצם, נסתכל בטיפוס שלו. תרשים העצם יכלול תחילה את שם הטיפוס על פיו נבנה העצם, את התכונות כפי שהן מוגדרות בטיפוס ואת הערכים שלהן. כלומר, את מצב העצם.

דוגמה לתרשים טיפוס ותרשים עצמים מן הטיפוס: מחשב

הטיפוס מחשב יוצג באופן הזה:

הטיפוס	מחשב
תכונות	בעלים מערכת הפעלה זיכרון האם דולק דיסק קשיח מיקום
פעולות	הדלק כבה הוסף זיכרון (_ כמה)

בהתאמה לייצוג הטיפוס יוצגו העצמים באופן הזה:

העצם	מחשב 1 של יפתח מטיפוס מחשב	העצם	מחשב 2 של יפתח מטיפוס מחשב	העצם	המחשב של מעין מטיפוס מחשב
תכונות וערכיהן	בעלים: יפתח מור מערכת הפעלה: חלונות XP זיכרון: Mg128 האם דולק: לא דיסק קשיח: 1G מיקום: חדר עבודה	תכונות וערכיהן	בעלים: יפתח מור מערכת הפעלה: חלונות Me זיכרון: Mg64 האם דולק: כן דיסק קשיח: 1G מיקום: משרד	תכונות וערכיהן	בעלים: מעין טל מערכת הפעלה: חלונות XP זיכרון: Mg128 האם דולק: לא דיסק קשיח: 2G מיקום: החדר של מעין

דוגמה לטיפוס: שולחן כתיבה

שולחן הכתיבה שלך הוא מאותו טיפוס כמו שולחנות הכתיבה של התלמידים בכיתתך, ויתכן אף שבבית הספר כולו ואפילו בבתי ספר אחרים. ניתן לומר שכל שולחנות הכתיבה הללו הם מאותו טיפוס. לכל שולחנות הכתיבה יש את התכונות הבאות: צבע המשטח העליון, מספר הרגליים, גובה

שולחן הכתיבה, מיקומו בחדר, שם התלמיד שיושב לידו ועוד. התכונות הן משותפות לכל שולחנות הכתיבה. יתכן שלתכונה מסויימת יהיו ערכים זהים בכל שולחנות הכתיבה, למשל מספר הרגליים. אך ברור שיש תכונות שערכיהן יהיו שונים משולחן לשולחן, למשל מיקום השולחן ושם התלמיד היושב לידו. גם הפעולות שניתן לבצע על שולחן כתיבה משותפות לכל העצמים מטיפוס שולחן כתיבה: אפשר להזיז כל שולחן כתיבה, בכל שולחן כתיבה ניתן להחליף את התלמיד היושב לידו, כל שולחן כתיבה אפשר לצבוע ועוד.

שולחן כתיבה	הטיפוס
גובה מספר רגליים צבע משטח עליון נמצא במקום שם תלמיד	תכונות
מה הגובה? עדכן גובה (_ גובה_ חדש) הזז את השולחן (_ לאן) החלף תלמיד (_ שם) צבע בצבע (_ צבע)	פעולות

שם ♥ : תרשים הטיפוס בדרך כלל לא כולל ערכי תכונות. ערך של תכונה יכול רק אם הוא מאפיין את כל העצמים מן הטיפוס ברגע בנייתם. ניתן להתייחס לכך כמו אל ברירת מחדל בזמן בניית העצמים. למשל, בדוגמה זו ניתן לתת לתכונה את ערך ברירת המחדל – 80 ס"מ.

תרגילים באקו"ן טיפוסים

כתוב תרשים

בעבור כל אחד מן הטיפוסים הבאים בצע:

- בחר לפחות 4 תכונות מאפיינות.
- בחר לפחות 6 פעולות מתאימות.
- הצג את הטיפוס בעזרת תרשים להצגת טיפוס.
- הצג שני עצמים בעזרת תרשים להצגת עצמים.



★ תרגיל 4: הטיפוס מכוננית



★ תרגיל 5: הטיפוס חלום



★ תרגיל 6: הטיפוס שיחת טלפון

חלק ב: תכנות עם עצמים

פרק זה התחיל בהגדרת המושג עצם. בהמשך הוגדר המושג טיפוס המתאר את המאפיינים המשותפים לקבוצת עצמים: תכונות ופעולות. לכל עצם יש את התכונות המוגדרות בטיפוס שלו וניתן להפעיל עליו את הפעולות המוגדרות בטיפוס שלו. עצם מתואר ע"י מצב העצם הכולל את הערכים של התכונות שלו. אך מהיכן מגיעים העצמים? מה משמעות המשפט "טיפוס הוא תבנית לבניית עצמים"?

בניית עצמים

בניית עצמים | כי טיפוס | או טיפוס קואזים | עצמים

כאשר ניגשים לפתרון בעיה בתכנות מונחה-עצמים יש לזהות תחילה את הטיפוסים הנדרשים לפתרונה ולתכנן את הפתרון בהתאם. אחרי זיהוי הטיפוסים, **מגדירים כל טיפוס כמחלקה בשפת התכנות.**

כאשר אנו אומרים שהטיפוס הוא **תבנית** אנו שמים דגש על כך שלא די בפיתוח טיפוסים כמחלקות בשפת התכנות. הגורמים הפועלים הם עצמים, וללא בניית עצמים אי אפשר לבצע שום פעולה. פעולות מתבצעות על עצמים. אם כך יש צורך לבנות עצמים. בניית העצמים תעשה במחלקה אחרת – המחלקה הראשית למשל. כדי לבנות עצם מטיפוס יש לבקש זאת במפורש מן המחלקה המייצגת אותו. בניית העצמים מתבצעת רק בשלב הרצה. בכל מקרה בניית עצמים מטיפוס אפשרית כמובן רק אחרי שהוגדרה המחלקה המייצגת את הטיפוס. בניית העצם תתבצע על ידי פעולה מיוחדת הנקראת **פעולה בונה**. תהליך בניית העצם נותן לעצם מזהה יחודי. ההבחנה בתכנות בין עצמים שונים תהיה על פי המזהה שלהם.

למרות שמודגש הצורך בבניית עצמים כדי להביא לביצוע של פעולות עליהם, חשוב מאד להבין את הרעיון שאפיון והגדרת טיפוס על ידי מחלקה בשפת התכנות עומד בפני עצמו. מחלקה מתארת טיפוס גם אם עדיין לא קיימים עצמים מהטיפוס. לטיפוס יש משמעות וקיום מעבר למשמעות ו לקיום של העצמים מן הטיפוס. ניתן לדבר על טיפוס מבלי להתייחס לעצמים מאותו טיפוס. למשל, מפעל לייצור צעצועים רוצה לייצר קו חדש של בובות דרקונים. תחילה יש לתכנן את מאפייני הבובות, כלומר מאפייני הטיפוס **בובת דרקונים**. התכונות: **חומר, גודל, סוג, ... הפעולות: הנעת רגליים, יריקת אש, ...** רק עם סיום הגדרות הטיפוס ניתן לדבר על בניית עצמים על פי קו הייצור התואם. חשוב להבדיל בין הטיפוס ובין העצמים מאותו טיפוס. טיפוס הוא המכנה המשותף לכל העצמים הבודדים מאותו טיפוס. עצם מטיפוס מסויים הינו מופע, דוגמה פרטית בתבנית הטיפוס. למשל, הטיפוס הוא בובת דרקונים, העצם הוא בובה מסויימת של הילד בן. לעצם יש תמיד את כל התכונות שהוגדרו בטיפוס שלו וכל התכונות מקבלות ערכים בזמן בניית העצם. כפי שתואר קודם חלקם יכולים להיות ערכי ברירת מחדל. בכל מקרה ניתן לשנות את ערכי התכונות לאחר שלב בניית העצם, על ידי הפעלת פעולות עליו.

הכּוּלָה הַבוּנָה

הפעולה הבונה כשמה היא – פעולה הבונה עצם. בכל טיפוס חייבת להיות פעולה בונה שתאפשר לבנות עצם. עצם חדש נבנה אך ורק על ידי הפעלה של הפעולה הבונה. הפעולה הבונה מחזירה עצם אשר לו התכונות בהתאם להגדרת הטיפוס שלו. הפעולה הבונה יכולה לקבוע ערכים לתכונות על ידי השמת קבועים, על ידי קלט, על ידי השמת ערכים מפרמטרים, או ערכי ברירת מחדל. הפעולה הבונה הסטנדרטית מבחינתנו תהיה זו המקבלת פרמטר המכיל ערך עבור כל תכונה של העצם. ניתן להגדיר בטיפוס אחד מספר פעולות בונות.

שים ♥ :

- ללא הפעלת פעולה בונה – אין עצם.
- פעולה בונה מזומנת רק פעם אחת עבור כל עצם.
- פעולה בונה מחזירה עצם.
- פעולה בונה אינה מופעלת על עצם.

דוגמה לזימון פעולה בונה עבור הטיפוס שולחן כתיבה

בתרשים הטיפוס נוסף את הפעולה הבונה. הפעולה הבונה המתוארת כאן מקבלת פרמטר עבור כל תכונה ובו הערך שיושם בתכונה של העצם המסוים שנבנה.

הטיפוס	שולחן כתיבה
תכונות	גובה: 80 ס"מ מספר רגליים צבע משטח עליון נמצא במקום שם תלמיד
פעולה בונה	בנה שולחן כתיבה (מספר רגליים, צבע משטח, מקום, שם תלמיד)
פעולות	מה הגובה? עדכן גובה (גובה_חדש) הזז את השולחן (לאן) החלף תלמיד (שם) צבע בצבע (צבע)

כאשר נפעיל את הפעולה הבונה, למשל כך :

בנה שולחן כתיבה(6, "כחול", "כיתה 30", "אדום") ← שולחן1

יתבצע התהליך הבא :

- על פי התכונות המוגדרות בטיפוס שולחן כתיבה, יוקצו שטחי זכרון עבור העצם.
- המזהה של העצם הוא : שולחן1.
- הערכים שהתקבלו כפרמטרים יושמו בהתאמה כערכים של התכונות.

ונקבל את העצם :

העצם	שולחן1 מטיפוס שולחן כתיבה
תכונות	גובה: 80 ס"מ מספר רגליים: 6 צבע משטח עליון: כחול נמצא במקום: כיתה 30 שם תלמיד: אדום

הפעלות פעולות על עצמם

על עצם אפשר להפעיל את כל הפעולות שהוגדרו בטיפוס שלו. לא כל הפעולות המוגדרות בטיפוס חייבות להתבצע, הפעולות יתבצעו על פי הנדרש לתהליך פתרון בעיה כלשהי. יש פעולות שתופעלנה בהרצה אחת, ואחרות שתופעלנה בהרצה אחרת. פעולה מסויימת יכולה להיות מופעלת שוב ושוב על אותו עצם, ברצף או שלא ברצף. לעומת זאת על עצם אחר לא תופעל כלל. למשל, אם הוגדר הטיפוס פחית שתייה, ונבנה על פיו העצם פחית שתייה 1, יתכן שאף אחד לא ישתה מן הפחית הזו, אך עדיין העצם קיים, וניתן על פי הגדרת הטיפוס של העצם להפעיל עליו את הפעולה שתה (_כמות), אך נבחר שלא להפעיל אותה. מאידך, אם נבנה עצם פחית שתייה 2, אפשר להפעיל עליו את הפעולה שתה (_כמות) כמספר הפעמים ששתו ממנה.

סוגי פעולות

על עצמים ניתן לבצע פעולות שונות. הגדרת הפעולות דומה מאד לפעולות שהגדרנו בעבר, אלא שכאן הפעולות מופעלות על עצמים.

קיימים שלושה הבדלים מרכזיים בשימוש בפעולות במסגרת גישת תכנות מונחה עצמים לבין השימוש בפעולות סטטיות שנעשה בספר עד כה. ההבדלים הם:

(1) הגדרת הפעולה – פעולות שהגדרנו עד כה שלא הופעלו על עצמים היו סטטיות (static), בעוד שפעולות המופעלות על עצמים אינן סטטיות;

(2) הדרך בה נזמן את הפעולה - זימון הפעולה יהיה על עצם: על עצם הפעל פעולה נדרשת;

(3) מאחר ופעולה מופעלת על עצם, ניתן לגשת בגוף הפעולה לתכונות של העצם (זאת בנוסף לגישה אל פרמטרים ומשתנים מקומיים של הפעולה). בהמשך כאשר נציג את מימוש העצמים בשפת התכנות נראה כיצד הבדלים אלו באים לידי ביטוי.

בפרק הגדרת פעולות אובחנו פעולות משני סוגים: פעולות המחזירות ערך ופעולות שלא מחזירות ערך. בתוך סיווג זה נעשה חלוקה נוספת באשר לפעולות על עצמים: פעולות מאחזרות – שמטרתן לאחזר ערך של תכונה, ופעולות מעדכנות - שמטרתן קביעת ערך של תכונה. הפעולות המאחזרות הן חלק מכלל הפעולות המחזירות ערך. הפעולות המעדכנות הן חלק מכלל הפעולות שלא מחזירות ערך. האיור הבא מתאר את אוסף הפעולות הניתנות להגדרה בטיפוס, ולהפעלה על עצם מטיפוס:



שים ♥ : הפעולה הבונה היא פעולה מיוחדת המחזירה עצם מן הטיפוס ואינה מופעלת על עצם. ולכן אינה כלולה כאן.

דוגמה לסיווג פעולות: תלמיד

הוגדר הטיפוס: ונבנה העצם:

העצם	תעודה 1 מטיפוס תעודה
תכונות	שם תלמיד: אמיר חירוני תעודת זהות: 123456789 ציון במתמטיקה: 90 ציון באנגלית: 74 ציון בלשון: 88

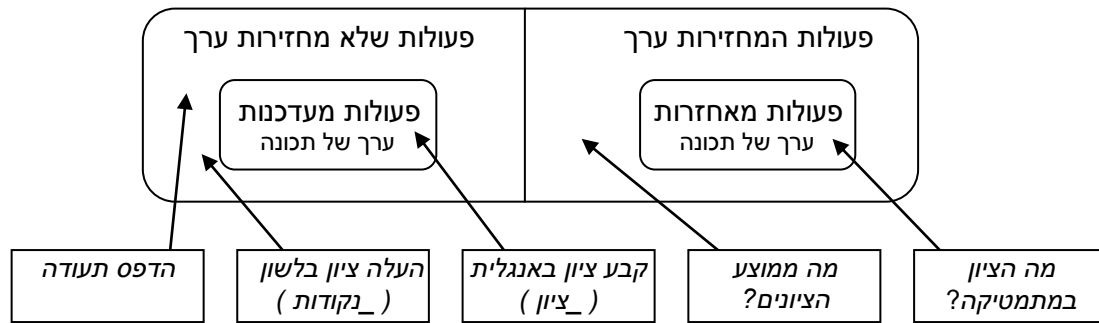
הטיפוס	תלמיד
תכונות	שם תלמיד תעודת זהות ציון במתמטיקה ציון באנגלית ציון בלשון
פעולות	מה הציון במתמטיקה? מה ממוצע הציונים? קבע ציון באנגלית (_ציון) העלה ציון במקצוע (_מקצוע , _נקודות) הדפס תעודה

הגדרת הפעולות היא הגדרה חלקית לצורך הדגמה בלבד. נתייחס אל משמעות הפעולות המוגדרות בטיפוס תעודה:

- הפעולה *מה הציון במתמטיקה?* היא פעולה המחזירה ערך ומוגדרת באופן מדויק יותר כפעולה מאחזרת. הערך שהיא מחזירה כאשר תופעל על העצם *תעודה1* הוא המספר *100*.
- הפעולה *מה ממוצע הציונים?* היא פעולה המחזירה ערך. הפעולה מחשבת את הערך הממוצע של הציונים על ידי פניה אל התכונות של העצם. הפעולה לא נקראת פעולה מאחזרת, כי התשובה המוחזרת מתקבלת מחישוב ולא מתקבלת ישירות מערך של תכונה. הערך שהיא מחזירה כאשר תופעל על העצם *תעודה1* הוא המספר *85*.
- הפעולה *קבע ציון באנגלית (_ציון)* היא פעולה שלא מחזירה ערך ומוגדרת באופן מדויק יותר כפעולה קובעת. היא קובעת ערך חדש לתכונה *ציון באנגלית* על פי הערך שמתקבל בפרמטר. לאחר הפעלת הפעולה *קבע ציון באנגלית (83)* על העצם *תעודה1* יהיה שינוי במצב העצם וערך התכונה *ציון באנגלית* יהיה *83*.
- הפעולה *העלה ציון במקצוע (_מקצוע , _נקודות)* היא פעולה שלא מחזירה ערך. הפעולה אינה נחשבת כפעולה קובעת, כי הערך החדש של התכונה מתקבל כתוצאה מחישוב ולא באופן ישיר מערך המתקבל כפרמטר. לאחר הפעלת הפעולה *העלה ציון במקצוע (לשון , 10)* על העצם *תעודה1* ישתנה ערך התכונה *ציון בלשון* מ-*88* ל-*98*.
- הפעולה *הדפס תעודה* היא פעולה שלא מחזירה ערך. פעולה זו מתייחסת אל ערכי התכונות ומדפיסה אותם בהתאם לצורת ההדפסה המוגדרת בפעולה. למשל הפעלת הפעולה *הדפס תעודה* על העצם *תעודה1*, ביחס למצב העצם הנתון, יכולה להיות:

תעודה	
שם התלמיד: אמיר חירוני	
תעודת זהות: 123456789	
מקצוע	ציון
מתמטיקה	90
אנגלית	74
לשון	88
ממוצע ציונים: 85	

ניתן לראות את שייכות חמש הפעולות המוגדרות בטיפוס תעודה אל קטגוריות הפעולות הכללית כפי שנראה באיור הבא:



הגדרת כעולה בסיס אל הכעות כעולה אל 38

קיים הבדל מהותי בין הגדרת פעולה לבין הפעלת הפעולה על עצם מסוים. בפרקי הספר הקודמים נעשתה הבחנה בין הגדרת פעולה – פיתוח אלגוריתם לביצוע מטרה, להפעלת פעולה – ביצוע של פעולה על פי הגדרתה. הגדרת פעולה אינה גורמת לביצוע. פעולה תתבצע רק על פי בקשה (קריאה/זימון) מפורשת לביצועה. כלל זה נשמר גם כאן. כדי להפעיל פעולה על עצם יש לציין את המזהה של העצם עליו רוצים להפעיל אותה ואת הפעולה שיש להפעיל. למשל, על העצם תעודה1, הפעל את הפעולה מה ממוצע הציונים?

תרגילים במיון כעולות



בעבור כל אחד מן הטיפוסים שהגדרת בתרגילים 4-6 בצע:

א. מייין את הפעולות שאפיינת לפי הקטגוריות: פעולות מאחזרות (ערך של תכונה), פעולות מעדכנות (ערך של תכונה), פעולות המחזירות ערך (ללא הפעולות המאחזרות), פעולות שלא מחזירות ערך (ללא הפעולות המעדכנות).

ב. הוסף פעולות כך שבכל אחת מן הקטגוריות יהיו לפחות שתי פעולות.

★ ★ תרגיל 7: הטיפוס מכונית

★ ★ תרגיל 8: הטיפוס חלום

★ ★ תרגיל 9: הטיפוס שיחת טלפון

סוג' מחלקות

מחלקה המ"צגת ג'טס אל מחלקה הראל'ת

מחלקה המייצגת טיפוס כוללת את התכונות של העצמים שייבנו מן הטיפוס ואת הפעולות שניתן יהיה להפעיל עליהם. אך היכן ייבנו העצמים? היכן תופעלנה הפעולות? כל אלה יוגדרו במחלקה הראשית. בפרקי הספר הקודמים הגדרנו מחלקה בה הוגדרה פעולה ראשית אחת (Main) ופעולות נוספות שהוגדרו לצורך פתרון הבעיה (שהיו סטטיות). הפעולה הראשית (Main) מנהלת את ביצוע האלגוריתם, והיא זו שמופעלת באופן אוטומטי בעת הרצת המחלקה. למחלקות מסוג זה נקרא **מחלקה ראשית**. מחלקה ראשית פותרת בעיה אלגוריתמית נתונה. בפעולה הראשית מגדירים משתנים נדרשים ומזמנים פעולות בסדר המתאים לצורך פתרון הבעיה. לעומת זאת מחלקה המייצגת טיפוס היא שונה. במחלקה המייצגת טיפוס מוגדרות התכונות והפעולות המאפיינות את העצמים שייבנו מן הטיפוס. במחלקה זו תוגדר גם הפעולה הבונה כדי לאפשר בנייה של עצם מן הטיפוס. בדרך כלל המחלקה כוללת גם הגדרה של פעולות מעדכנות ופעולות מאחזרות לכל תכונה זאת כדי לשמור על עקרון הסתרת המידע. גישה לתכונות מחוץ למחלקה המייצגת את הטיפוס תהיה רק דרך פעולות מורשות (קובעות ומאחזרות). לא מקובל לגשת לתכונות באופן ישיר (בשפת התכנות יוגדרו התכונות עם בקרת גישה private כך שהן יחסמו לגישה מוץ למחלקה בה הן מוגדרות). החל משלב זה נעשה הבחנה בין מחלקה המייצגת טיפוס למחלקה הראשית. כאשר ניגש לפתרון בעיה, נבחן תחילה האם לצורך פתרון מתאים להשתמש בעצמים ואם כן - מה הטיפוס שלהם (ניתן להגדיר גם מספר טיפוסים). אם יש צורך בהגדרת טיפוסים, תוגדרנה המחלקות המתאימות לייצוג הטיפוסים, על פי השלבים שיפורטו בסעיף הבא. בנוסף תוגדר המחלקה הראשית, בה יבנו העצמים הנדרשים לצורך פתרון הבעיה, ותופעלנה עליהם הפעולות הנדרשות בהתאם לסדר הנדרש. תפקיד המחלקה הראשית הוא לבצע את האלגוריתם. המחלקה הראשית אינה מייצגת טיפוס ולכן לא נבנה ממנה עצמים. אין משמעות לעצמים כאלה, גם אם כפי שנראה בהמשך בנייתם בשפת התכנות אפשרית תמיד. בפתרון בעיה בגישת תכנות מונחה עצמים יש הגדרה של מספר מחלקות. מספר מחלקות מהוות יחד פרויקט המשמש לצורך פתרון בעיה. פרויקט יכול להכיל מספר מחלקות המייצגות טיפוסים מהם נבנה עצמים, ויכיל בנוסף מחלקה ראשית אחת, בה תוגדר הפעולה הראשית ובה יתבצע האלגוריתם הראשי לפתרון הבעיה.

חלוקת כ'ה ל'טוס'ס - ע'קרון המודולר'ות

חלוקת בעיה לטיפוסים המשתתפים בה ופיתוח מחלקה לייצוג כל אחד מן הטיפוסים הוא חלק מעיקרון המודולריות בתכנות מונחה עצמים. עקרון המודולריות בא לידי ביטוי בשני מימדים עיקריים. האחד, חלוקת בעיה למודולים - במקרה זה טיפוסים - מאפשרת פיתוח ובדיקה יסודיים של כל אחד בנפרד. השני, ניתן לפתח מחלקה לצורך פתרון בעיה אחת, אך להשתמש בה שוב לפתרון בעיות אחרות. אחרי שפיתחנו מחלקה ובדקנו אותה, אפשר להשתמש בה גם בהרבה הקשרים אחרים, ולצורך פתרון בעיות שונות מבלי צורך לשוב ולאפיין אותה או לשוב ולפתח את הקוד שלה בשפת התכנות. כל מה שיש לעשות הוא לשייך את המחלקה לפרוייקט חדש בהתאם לכללי שפת התכנות ולפעמים כללי סביבת הפיתוח. לאחר שמחלקה משוייכת לפרוייקט ניתן לבנות על פיה עצמים ולהשתמש בהם ככל שנדרש לצורך פתרון בעיה.



חלק ג: פיתוח מחלקה המייצגת טיפוס

מחלקה המייצגת טיפוס כוללת את החלקים הבאים: כותרת המחלקה, הגדרת תכונות, הגדרת פעולות בונות, הגדרת פעולות. לפניך תיאור כללי של מבנה מחלקה:

```
public class < שם_הטיפוס >
1 {
    // תכונות           2
    // פעולות בונות     3
    // פעולות           4
}
```

מבנה מחלקה המייצגת טיפוס

הסבר על מרכיבי המחלקה (בהדגמה על הטיפוס תלמיד - Student)

הטיפוס	תלמיד
תכונות	שם תלמיד תעודת זהות ציון במתמטיקה ציון באנגלית ציון בלשון
פעולות	מה הציון במתמטיקה? מה ממוצע הציונים? קבע ציון באנגלית (_ ציון) העלה ציון במקצוע (_מקצוע, _נקודות) הדפס תעודה

1 **כותרת המחלקה:** שם המחלקה מתחיל במילים **public class** ולאחריו שם המחלקה. נהוג ששם מחלקה מתחיל באות גדולה. שם הקובץ בו נשמרת המחלקה צריך להיות זהה לשם המחלקה, ועם סיומת **.cs**. לדוגמה בטיפוס תלמיד:

```
public class Student
{
    .....
}
```

2 הגדרת התכונות

בתחילת הגדרת המחלקה נרשמות התכונות. הגדרה של כל תכונה כוללת את הטיפוס שלה (שלם, ממשי, בוליאני, מערך, וכו') ואת שם התכונה. כללי הגדרת התכונות הם כמו הכללים עבור הגדרת משתנים. התכונות יוגדרו עם בקרת גישה **private**. הסבר על כך ניתן בהמשך. שים לב: ♥ : המבחין בין הגדרת התכונות להגדרת משתנים הוא במיקום ההגדרה וגם במשמעות:

(1) משתנים מוגדרים בתוך גוף פעולה, בעוד שתכונות מוגדרות בתוך גוף המחלקה.

(2) משתנים הם מקומיים לפעולה ולא מתקיימים עם סיומה. תכונות הן חלק מן העצם ומתקיימות לפני זימון הפעולה וגם אחרי סיומה. לדוגמה בטיפוס תלמיד:

```
private string name;
private string idNum;
private int math;
private int english;
private int language;
```



הפעולה הבונה היא פעולה החייבת להיות מוגדרת בכל טיפוס, בלעדיה לא ניתן לבנות עצם. הפעולה הבונה מופעלת כאשר רוצים לבנות עצם חדש. שים ♥ : פעולה זו תופעל רק פעם אחת עבור כל עצם. **לפעולה הבונה יש שם זהה לשם המחלקה המייצגת את הטיפוס**. לפעולה זו לא מוגדר טיפוס הערך המוחזר מאחר וידוע שהיא מחזירה עצם מן הטיפוס המוגדר (לכן לא מופיע בה משפט return). ניתן להגדיר מספר פעולות בונות באותה מחלקה השונות זו מזו בכותרת שלהן. כותרת של פעולה כוללת את שם הפעולה (שבמקרה זה הוא זהה) ואת רשימת הפרמטרים שלה. כלומר הבדל בין פעולות בונות יבוא לידי ביטוי ברשימת הפרמטרים. כל שינוי (כמות פרמטרים, טיפוסים שונים, סדר הפרמטרים) המאפשר הבחנה ברורה בין הפעולות הבונות מאפשר הגדרה של פעולה בונה נוספת.



מבנה פעולה בונה

```

(שמות הפרמטרים וטיפוסיהם) שם_הפעולה_כשם_הטיפוס public
{
    // הוראות לביצוע
}

```

ביצוע פעולה בונה כולל ארבעה שלבים:

- שלב 1: הגדרת שיטחי זיכרון מתאימים לעצם החדש על פי הגדרת התכונות בטיפוס שלו.
- שלב 2: איתחול התכונות על פי ערכי ברירת המחדל של הטיפוסים שלהן.
- שלב 3: ביצוע גוף הפעולה הבונה.
- שלב 4: החזרת העצם למקום ממנו זומנה הפעולה הבונה. בשלב זה נדבר על השמה לתוך מזהה העצם.

הפעולה הבונה קובעת ערכים לתכונות העצם. כדי שהפעולה הבונה תקבע ערכים לתכונות בהתאם לביצוע אלגוריתם היא יכולה לקבל את הערכים בעזרת פרמטרים. כלומר הפעולה הבונה יכולה לקבל פרמטר המכיל ערך עבור כל תכונה שאת ערכה היא רוצה לקבוע. ערכי תכונות יכולים להתקבל גם על ידי השמה של קבוע לתכונה או על ידי קלט ואז אין צורך בפרמטר. אם אין קביעה של ערכים לתכונות על פי פרמטרים או על פי קבועים, נשאר בהם הערך של ברירת המחדל לפי הטיפוס של התכונה שהושם בשלב 2. לדוגמה הפעולה הבונה הבאה בטיפוס תלמיד:

<pre> public Student (string n, string id) { name = n; idNum = id; } </pre>	<ul style="list-style-type: none"> שם הפעולה כשם הטיפוס – בדיוק! הפעולה מקבלת שני פרמטרים: האחד עבור הערך שיושם לתכונה שם ונקרא n, השני עבור הערך שיושם לתכונה ת.ז. ונקרא id. גוף הפעולה כולל שתי הוראות השמה לשתי תכונות. משמעות ההוראה: name = n; היא: השם לערך התכונה name את הערך המתקבל מן הפרמטר n.
---	---

הסבר ביצוע הפעולה:

- הפעולה הבונה תזומן באופן בו מזמנים פעולות המחזירות ערך, מאחר והיא מחזירה עצם מן הטיפוס (הסבר מפורט בסעיף הגדרת עצמים והפעלת פעולות עליהם).
 - עם זימון הפעולה הבונה, מעיד שם הפעולה מאיזה טיפוס צריך להיות העצם שנבנה.
 - על פי שלב 1: יש פנייה אל הטיפוס כדי לראות אילו תכונות מוגדרות בו, ומוקצים שטחי זיכרון מתאימים לתכונות העצם החדש. למשל: תכונה בשם `name` מטיפוס `string`, ותכונה בשם `math` מטיפוס `int`.
 - על פי שלב 2: מושמים לתכונות ערכי ברירת המחדל על פי טיפוסיהם. למשל: לתכונה `name` מושם הערך `null` (כי המחרוזת טרם אותחלה), ולתכונה `math` מושם הערך השלם 0 (לפי ברירת המחדל עבור הטיפוס `int`).
 - על פי שלב 3: מתבצע גוף הפעולה הבונה. כלומר, מתבצעות שתי הוראות ההשמה הקובעות את ערכי התכונות: `name`, `idNum`. יתר התכונות נשארות עם ערכי האתחול על פי ברירת המחדל שהתקבלו בשלב 2.
 - על פי שלב 4: מוחזר עצם חדש אל המקום ממנו התבקשה הבניה שלו. העצם מוחזר כערך אחד – כמוסה (פירוט בהמשך הפרק).
 - ערכי התכונות `name` ו-`idNum` עודכנו על פי הערכים שהתקבלו בפרמטרים, ערכי התכונות האחרות `math`, `english`, `language`, הם כערך ברירת המחדל של הטיפוס שלהן `int` – הערך 0.
- הגדרה זהה לחלוטין לפעולה בונה זו היא:

```
public Student (string name, string idNum)
{
    this.name = name;
    this.idNum = idNum;
}
```

דרך הגדרה זו מקובלת בתכנות מונחה עצמים. כאשר מאתחלים ערכים של תכונות, מקובל לקרוא גם לפרמטרים בשם זהה לשם התכונה. מצב זה לא אפשרי מבחינת שפת התכנות מאחר ולא יתכן למשל ש `name` ישמש באותה פעולה במקביל כשם פרמטר וגם כשם תכונה. המילה השמורה `this` מציינת שהכוונה לתכונה של העצם הזה – **העצם הנוכחי** אליו מתייחסת הפעולה. כלומר `this.name` מתייחס לתכונה של העצם, בעוד `name` הוא הפרמטר. בספר זה נשתמש בעצם הנוכחי רק בהקשר כזה. ניתן להוסיף התייחסות לעצם הנוכחי (`this`) בכל פעם שפונים לתכונה או לפעולה המוגדרות בטיפוס מתוך אותו הטיפוס.

נראה דוגמה לפעולה בונה נוספת היכולה להיות מוגדרת בטיפוס תלמיד. פעולה בונה זו יכולה להופיע בנוסף לפעולה הבונה שתוארה לעיל מאחר והכותרת שלה שונה. פעולה בונה זו מקבלת פרמטרים עבור כל אחד מערכי התכונות של הערך החדש שנוצר:

```
public Student (string name, string idNum, int math, int english, int language)
{
    this.name = name;
    this.idNum = idNum;
    this.math = math;
    this.english = english;
    this.language = language;
}
```

דוגמאות נוספות לפעולות בונות ומשמעותן

פעולה בונה	הסבר
<pre>public Student () { }</pre>	<p>פעולה בונה זו היא פעולה בונה כמו קודמותיה. הפעולה ניתנת להגדרה בנוסף לפעולות הקודמות משום שהכותרת שלה שונה – אין לה פרמטרים. היא תבנה ותחזיר עצם שכל הערכים של התכונות שלו נקבעו על פי ברירת המחדל. פעולה בונה זו נקראת פעולה בונה ברירת מחדל.</p>
<pre>public Student (string n, string id) { name = n; idNum = id; math = 100; english = 100; language = 100; }</pre>	<p>הפעולה קובעת את הערכים של הציונים להיות תמיד 100. שים ♥ : פעולה בונה זו <u>לא</u> יכולה להיות מוגדרת במחלקה אם מוגדרת בה גם הפעולה הבונה שניתנה בדוגמה הראשונה. לשתיהן כותרת זהה ולכן אינן יכולות להיות מוגדרות יחד במחלקה.</p>

שים ♥ : אם בהגדרת מחלקה המייצגת טיפוס לא תוגדר כלל פעולה בונה, המערכת מתייחסת כאילו הייתה מוגדרת הפעולה בונה ברירת מחדל. אם רוצים אפשרות להשתמש בפעולה בונה ברירת מחדל גם כאשר מוגדרות פעולות בונות אחרות, אזי יש להגדיר אותה במפורש. במקרה וכבר מוגדרת פעולה בונה אחת לפחות, לא ניתן להשתמש בהגדרה האוטומטית של פעולה בונה ברירת מחדל.

הגדרת הכאולות

4

להלן נתייחס לסוגי הפעולות השונים המוגדרים בטיפוס : פעולות מאחזרות, פעולות קובעות, ופעולות חישוביות ביניהן כאלה המחזירות ערך וכאלה שלא.

פעולות מאחזרות

פעולה מאחזרת היא פעולה המאחזרת ערך של תכונה. מקובל להגדיר לכל תכונה פעולה מאחזרת. כל מה שפעולה זו עושה הוא להחזיר את ערך התכונה. פעולה מאחזרת לא מקבלת פרמטרים. מקובל ששם פעולה מאחזרת יכלול את המילה Get ואת שם התכונה שאת הערך שלה היא מחזירה. גוף הפעולה יכלול רק משפט **return** המחזיר את ערך התכונה המתאימה.



מבנה פעולה מאחזרת

```
public <שם_התכונה> Get<טיפוס_התכונה>()
{
    return <שם_התכונה>
}
```

דוגמה

```
public string GetName()
{
    return name;
}

public int GetMath()
{
    return math;
}
```


פעולות קובעות

פעולה קובעת היא פעולה המאפשרת לקבוע ערך חדש לתכונה. מקובל להגדיר לכל תכונה פעולה קובעת. כל מה שפעולה זו עושה הוא לקבל ערך חדש לתכונה מן הפרמטר ולהשים אותו בתכונה. פעולה קובעת לא מחזירה שום ערך. מקובל ששם פעולה קובעת יכלול את המילה Set ואת שם התכונה שאת הערך שלה היא קובעת. גוף הפעולה יכלול רק משפט השמה לתכונה המתאימה.

```
public void Set<פרמטר>(<טיפוס_התכונה> <שם_התכונה>)  
{  
    <פרמטר> = <שם_התכונה>;  
}
```

דוגמה

```
public void SetName(String name)  
{  
    this.name = name;  
}  
public void SetMath(int math)  
{  
    this.math = math;  
}
```

**מבנה פעולה קובעת**

פעולות מאחזרות ופעולות קובעות – לשם מה?

אחת המטרות המרכזיות של תכנות מונחה עצמים היא לאפשר הגדרה של טיפוסים שיוכלו לשמש לפתרון של בעיות שונות - מודולריות. אם נגדיר טיפוס באופן מלא על פי תפקידו, אזי אפשר יהיה להשתמש בו בהקשרים שונים ללא צורך לתכנת אותו מחדש. כדי לאפשר שימוש כזה ממקומות שונים, על מפתח של תכנית חדשה (פרויקט חדש) יהיה לדעת רק את כותרות הפעולות המוגדרות במחלקה ואת משמעותן – ממשק הפעולות. על מפתח התכנית יהיה לדעת "מה" הפעולות עושות ולא "כיצד" הן מבצעות זאת. מקובל לא לאפשר למחלקות אחרות לפנות באופן ישיר אל תכונות המוגדרות במחלקה אחרת. אי לכך יש לכתוב פעולות שמחד יאפשרו לקבל ערך של תכונות – **פעולות מאחזרות**, ומאידך יאפשרו קביעה של ערכים חדשים לתכונות – **פעולות קובעות**.

כאשר רוצים למנוע במפורש פנייה אל תכונות ממחלקות אחרות מזו בה הן מוגדרות, יש להגדיר את התכונות עם בקרת גישה `private`. למשל: `private string name;` בספר זה לא נרחיב בנושא בקרת גישה לתכונות ו/או פעולות מעבר להכרח הנדרש. התכונות במחלקות שנפתח יהיו תמיד `private`.

פעולות חישוביות

פעולה חישובית היא כל פעולה שאינה בונה, אינה מאחזרת ואינה קובעת. פעולה חישובית יכולה להחזיר ערך או לא. בין פעולות אלו יש המחשבות ערכים (כמו חישוב ממוצע ציונים), ויש שלמשל מדפיסות (כמו הדפסת תעודה). לדוגמה עיין בשלוש הפעולות החישוביות המוגדרות בטיפוס תלמיד:

מטרת הפעולה	הפעולה
1. הפעולה מחשבת ומחזירה את הערך השלם של ממוצע הציונים במתמטיקה, אנגלית ולשון:	<pre>public double ComputeAverage() { return (math + english + language) / 3.0; }</pre>
2. הפעולה מקבלת שם מקצוע subject ומספר נקודות	<pre>public void AddPointsToSubject(string subject, int points) { if (subject.Equals("math"))</pre>

<pre> math = math + points; else if (subject.Equals("english")) english = english + points; else language = language + points; } </pre>	<p>points ומעלה את ציונו של התלמיד במקצוע subject ב- points נקודות:</p>	
<pre> public void PrintCertificate() { Console.WriteLine(" Certificate "); Console.WriteLine("====="); Console.WriteLine(" Name: " + name); Console.WriteLine(" Id: " + idNum); Console.WriteLine(); Console.WriteLine(" Math: " + math); Console.WriteLine(" English: " + english); Console.WriteLine(" Language: " + language); Console.WriteLine(); Console.WriteLine(" The average grade: " + computeAverage()); } </pre>	<p>3. הפעולה מדפיסה לתלמיד תעודה:</p> <p>שים ♥ : הפעולה מזמנת פעולה אחרת שהוגדרה במחלקה, את הפעולה ComputeAverage().</p>	

מחלק מילולי אתיאור כעולות לל ט'כוס

לצורך תיאור מפורט של פעולות הכלולות בטיפוס משתמשים בממשק מילולי המתאר אותן. כל פעולה מתוארת בממשק על ידי: (1) כותרת הפעולה הכוללת את שמה ואת רשימת הפרמטרים שלה בעברית, (2) תיאור מילולי של מה הפעולה מקבלת, מה היא מבצעת ומה היא מחזירה. לדוגמה ממשק מילולי המתאר חלק מן הפעולות בטיפוס תלמיד:

הפעולה	תיעוד הפעולה
בנה-תלמיד (שם, תז)	פעולה בונה המקבלת את שם התלמיד ואת מספר תעודת הזהות שלו, וקובעת בהתאמה את ערכי התכונות שלו
אחזר-ציון-במתמטיקה ()	פעולה המחזרת את הציון במתמטיקה
קבע-ציון-באנגלית (ציון)	פעולה הקובעת את הציון באנגלית להיות ציון
חשב-ממוצע-ציונים ()	פעולה המחשבת ומחזירה את ממוצע הציונים של תלמיד
העלה-ציון-במקצוע (מקצוע, נקודות)	פעולה המקבלת מקצוע כמחרוזת ו נקודות ומעלה את הציון של התלמיד במקצוע במספר נקודות
הדפס-תעודה ()	פעולה המדפיסה לתלמיד תעודה

מחלק בלכת התכנות אתיאור כעולות לל ט'כוס

בהתאמה לממשק המילולי ניתן לכתוב ממשק בשפת התכנות. בממשק בשפת התכנות תיכתב עבור כל פעולה הכותרת שלה בשפת התכנות. כותרת פעולה מורכבת מהשם שלה, רשימת הפרמטרים וטיפוס הערך המוחזר. לדוגמה ממשק בשפת התכנות התואם לממשק המילולי שתואר לעיל:

תיעוד הפעולה	כותרת הפעולה
פעולה בונה המקבלת את שם התלמיד ואת מספר תעודת	<code>public Student(string n, string id)</code>



	הזהות שלו, וקובעת בהתאמה את ערכי התכונות שלו
<code>public int GetMath()</code>	פעולה המאחזרת את הציון במתמטיקה
<code>public void SetEnglish(int eng)</code>	פעולה הקובעת את הציון באנגלית להיות <code>eng</code>
<code>public int ComputeAverage()</code>	פעולה המחשבת ומחזירה את ממוצע הציונים של תלמיד
<code>public void AddPointsToSubject(string subject, int points)</code>	פעולה המקבלת מקצוע <code>subject</code> כמחרוזת ונקודות <code>points</code> ומעלה את הציון של התלמיד במקצוע על פי מספר הנקודות
<code>public void PrintCertificate()</code>	פעולה המדפיסה לתלמיד תעודה

חלק ד: פיתוח מחלקה ראשית המשתמשת בעצמים

את הגדרת העצמים והפעלת הפעולות עליהם לא נבצע במחלקה המגדירה את הטיפוס אלא במחלקה אחרת. בשלב זה נגדיר עצמים במחלקה הראשית ושם גם נפעיל עליהם פעולות. יש לזכור כי שפת התכנות מאפשרת לעשות גם דברים אחרים, אך בספר זה ינתן דגש על מבנה מומלץ, בהיר ומסודר של פרויקט בתכנות מונחה עצמים. נגדיר מחלקה ראשית ובה פעולה ראשית. בפעולה הראשית (Main) של המחלקה הראשית נגדיר עצמים ונפעיל עליהם פעולות.

הגדרת צמ"ס

כדי שנוכל לבצע פעולות המוגדרות במחלקה המייצגת יישות שהוגדרה, עלינו תחילה לבנות עצמים. פעולות המוגדרות במחלקה כזו מופעלות רק על עצמים מן הטיפוס. לצורך בניית העצמים מוגדרת הפעולה הבונה, הבונה עצם על פי הגדרת הטיפוס. לכל עצם שנבנה יש מזהה המאפשר לפנות אליו בהמשך התכנית (משתנה). מבנה הצהרה על עצם, קביעת המזהה שלו ובנייה שלו מתואר באיור הבא:



מבנה הצהרה ובנייה של עצם

```
<זימון_פעולה_בונה> new = <מזהה_העצם> <טיפוס_העצם>
```

א

ב

ג

ד

הסבר על מרכיבי הצהרה ובנייה של עצם

- א - טיפוס העצם. כמו בהצהרה על כל משתנה המתחילה בהגדרת הטיפוס שלו, כך גם כאן. העובדה שמדובר בטיפוס אשר הוגדר על ידי המשתמש אינה משנה דבר.
- ב - מזהה העצם הוא המשתנה המייצג את העצם שנבנה. שם המשתנה בהתאם לכל הצהרה אחרת על משתנה.
- ג - המילה השמורה `new`, משמשת לבנייה של עצמים חדשים. תמיד כשרוצים לבנות עצם חדש יש לכתוב את המילה `new`.
- ד - זימון הפעולה הבונה. הזימון כולל את שם הפעולה הבונה, הזוהה לשם המחלקה המייצגת את הטיפוס, ואת רשימת הפרמטרים המתאימה. אם יש מספר פעולות בונות אזי הפעולה הבונה תיבחר בהתאם לכותרת שלה.

לדוגמה הצהרה ויצירה של עצם מן הטיפוס תלמיד :

```
Student stu1 = new Student("Asaf Cohen", "123456789", 90, 80, 92);
```

א ב ג

ד

לאחר ביצוע הוראה זה קיים עצם מטיפוס Student שהמזהה שלו בתכנית הוא `stu1`, והתכונות שלו הן : שם- אסף כהן, ת.ז. - 123456789, ציון במתמטיקה- 90, ציון באנגלית- 80, ציון בלשון- 92. הערך של המשתנה `stu1` הוא הפנייה אל העצם מטיפוס Student שנוצר. לאחר יצירת העצם ניתן להפעיל עליו פעולות.

צמ כערק לל מלתנה - ע'קרון ההכמסה (encapsulation)

כתוצאה מביצוע הפעולה הבונה נבנה עצם בזיכרון וההפנייה אליו היא הערך המושם למשתנה מן הטיפוס המתאים - מזהה העצם. ערך זה הוא כמו כל ערך של טיפוס אחר. משתמשים במונח **כמוסה** (כמו כמוסה המכילה תרופה) כדי לחזק את המשמעות של היות העצם יחידה אחת. אמנם ניתן לגשת למרכיבים של העצם - אל התכונות שלו - אך הוא כולו יחידה אחת.

העלות העולות ע' צמ'ים

כאשר יש עצם ניתן להפעיל עליו פעולות שמוגדרות בטיפוס שלו. הפעלת הפעולה היא בתחביר זהה לתחביר הפעלת פעולות שכבר הכרנו - למשל בהפעלת פעולות על עצמים מטיפוס `string`. כאשר הגדרנו עצם מטיפוס `string` היה לו מזהה, ועליו הפעלנו, ע"י שימוש בתו נקודה, פעולות שהוגדרו בטיפוס שלו. במקרה של הטיפוס `string` מדובר בטיפוס שמוגדר מראש בשפת Java וכך גם הפעולות שלו. כאן נפעיל את אותו התהליך על עצמים מטיפוס שהמשתמש הגדיר. מבנה הפעלה של פעולה על עצם מתואר במבנה הבא :

< זימון_הפעולה > . < מזהה_העצם >

א ב ג

הסבר על מרכיבי הפעלת פעולה על עצם

- א - מזהה העצם עליו רוצים להפעיל את הפעולה.
 - ב - התו '.', המסמן הפעלת פעולה על העצם.
 - ג - רישום כותרת הפעולה בהתאם להגדרתה בטיפוס של העצם.
- משמעות המשפט :** על העצם שהמזהה שלו הוא <מזהה_העצם> הפעל את הפעולה <זימון_פעולה>.
- שים ♥ : המבנה מתאר זימון כללי של פעולה על עצם. אם הפעולה המוזמנת היא פעולה שלא מחזירה ערך אזי למבנה הזימון יש להוסיף ' ;' לסוף המשפט. אם הפעולה היא פעולה שמחזירה ערך אזי יש "לעשות משהו" עם ערך זה כרגיל בפעולות המחזירות ערך - להשים למשתנה אחר, או להדפיס את הערך המוחזר, וכו'.

לדוגמה הפעלת הפעולה העלה ציון במקצועל_מקצוע, (נקודות) על העצם `stu1` :

```
stu1.AddPointsToSubject("Math", 5);
```

א ב ג

משמעות ההוראה : על העצם `stu1` הפעל את הפעולה : `AddPointsToSubject("Math", 5)` : נסתכל על דוגמה מלאה למחלקה ראשית הבודקת את הטיפוס Student :



```

public class TestStudent
{
    // הפעולה הראשית
    public static void Main(string[] args)
    {
        Student stu1 = new Student("Asaf Cohen", "123456789", 90, 80, 92);
        stu1.AddPointsToSubject("Math", 5);
        stu1.PrintCertificate();

        Student stu2 = new Student("Yael Mor", "111222333");
        stu2.SetMath(70);
        stu2.SetEnglish(100);
        stu2.SetLanguage(95);
        Console.WriteLine("The average of " + stu2.GetName() + " is: " +
            stu2.ComputeAverage());
    }
}

```



בחלק הראשון מוצהר ונבנה העצם `stu1`. על העצם מופעלת הפעולה `העלה-ציון-במקצוע` עם הפרמטרים מתמטיקה ומספר נקודות 5. לאחר מכן מודפסת תעודה ל `stu1`. בחלק השני מוצהר ונבנה העצם `stu2`, הפעם תוך שימוש בפעולה `בונה עם כותרת שונה`. במקרה זה לאחר בניית העצם ערכי הציונים שלו הם 0, `כערכי ברירת המחדל`. לאחר מכן מופעלות על העצם שלוש פעולות קובעות, כל אחת קובעת ציון במקצוע אחר. לאחר מכן מודפסת הודעה הכוללת את שם התלמיד תוך שימוש בפעולה `המאחזרת את שמו`, את הממוצע שלו תוך שימוש בפעולה `המחזירה את הממוצע`.

מה בין פעולות המוגדרות כ- static לכאלה שלא

עד לפרק זה עסקנו במימוש מחלקות ובהן פעולות סטטיות. המחלקות שהגדרנו עד לפרק זה לא היו מחלקות שייצגו טיפוס. בפרק זה למדנו כיצד להגדיר מחלקה המייצגת טיפוס, וכל הפעולות שהוצגו לא הוגדרו כ- `static`. ההבחנה המהותית היא שפעולות שאינן מופעלות על עצמים מוגדרות כ- `static` בעוד פעולות המופעלות על עצמים לא מוגדרות כ- `static`. מכאן נובע כי כל הפעולות המוגדרות במחלקה המייצגת טיפוס, ומיועדות להיות מופעלות על עצמים לא יוגדרו כ- `static`.

מלמאות הטיפוס כתבנית ליצירת צממים

בתחילת הפרק הודגש כי טיפוס הוא תבנית ליצירת עצמים. הגדרת הטיפוס מבחינת המבנה שלה כוללת את הגדרת התכונות, את הגדרת הפעולות הבונות עצמים, ואת הגדרת הפעולות האחרות המופעלות על עצמים. לאחר הגדרת מחלקה המייצגת יישות "לא קורה כלום". כדי שמשהו יתבצע יש תחילה לייצר עצמים. גם לאחר שיש עצמים "לא קורה כלום" (פרט להקצאת זיכרון עבור התכונות) אם לא נפעיל עליהם פעולות. הפעלת הפעולות על העצמים תהיה על פי הצורך המתאים לפתרון הבעיה בה אנו עוסקים. כל אחת מן הפעולות שהוגדרו בטיפוס ניתנות להפעלה על כל עצם, אך זה לא אומר שכולן תופעלנה תמיד. חלק מן הפעולות יכולות לשמש לפתרון בעיה אחת, וחלק אחר לפתרון בעיה שונה. כל פעולה יכולה להתבצע מספר פעמים ככל שנדרש לפתרון הבעיה.

האלגוריתם הראשי, שיפתור בעיה המשתמשת בטיפוס שהוגדר על ידי המשתמש, יופיע במחלקה הראשית ובפעולה הראשית. הפתרון יכול לשלב גם פעולות נוספות שתוגדרנה במחלקה הראשית.

אלגוריתם האלתאם בעצמים

- אלגוריתם מילולי בעברית ישתמש בממשק הפעולות בעברית ויותאם לפעולות הקשורות בעצמים בניית עצמים והפעלת פעולות עליהם. נשתמש בשני ניסוחים חדשים:
 - עבור בניית עצם נשתמש במילים "בנה עצם חדש ע"י"
 - עבור הפעלת פעולה על עצם נשתמש במילים "על העצם ... הפעל את הפעולה..."
- לדוגמה אלגוריתם התואם את הפעולה הראשית המוגדרת במחלקה TestStudent:
 - (1) בנה עצם חדש ע"י בנה-תלמיד ("אסף כהן", "123456789", "90, 80, 82) והשם אותו במשתנה stu1
 - (2) על העצם stu1 הפעל את הפעולה העלה-ציון-במקצוע ("מתמטיקה", 5)
 - (3) על העצם stu1 הפעל את הפעולה הדפס-תעודה



דוגמה
פתורה

דוגמה כתורה מלאה: שירים

סעיף זה מציג פתרון מלא למימוש הטיפוס שיר (Song). תחילה יוצג תיאור מילולי של הטיפוס, אחר כך תוצג המחלקה המממשת את הטיפוס, ואחר כך תוצג המחלקה הראשית בה ייבנו עצמים מן הטיפוס ויופעלו עליהם פעולות.

תיאור הטיפוס שיר

הטיפוס שיר (Song) מאפיין שירים ומתייחס אל שלוש תכונות: שם השיר, שם המבצע ואורך השיר בשניות. ניתן כמובן להוסיף תכונות נוספות. האיור שלהלן מתאר את הטיפוס שיר וכולל הרחבה בסיווג הפעולות. כל הפעולות המופיעות באיור יופיעו בהגדרת המחלקה המממשת את הטיפוס:

הטיפוס	שיר
תכונות	שם השיר שם המבצע אורך השיר בשניות
פעולות	בנות שיר (שם, מבצע, אורך) שיר () { פעולה בונה הקולטת ערכים לתכונות }
	מאחזרות אחזר שם שיר () אחזר שם מבצע () אחזר אורך ()
	קובעות קבע שם שיר (שם 1) קבע שם מבצע (מבצע 1) קבע אורך (אורך 1)
	חישוביות הגדל אורך שיר ב (מספר_שניות) הקטן אורך שיר ב (מספר_שניות) סיווג אורך () { קצר- קטן מ-2 דקות, ארוך- ארוך מ-4 דקות, ממוצע - אחרת } הדפס פרטי שיר()


```
using System;
public class Song
{
```

```
    private string name;           // Name of song
    private string performer;      // Name of the Performer of the song
    private int length;           // Length of song in seconds
```

תכונות

```
public Song(string name, string performer, int length)
{
```

```
    this.name = name;
    this.performer = performer;
    this.length = length;
```

פעולות
בונות

```
public Song()
{
```

```
public string GetName()
{
```

```
    return name;
```

```
public string GetPerformer()
{
```

```
    return performer;
```

```
public int GetLength()
{
```

```
    return length;
```

פעולות
מאחזרות

```
public void SetName(string name)
{
```

```
    this.name = name;
```

```
public void SetPerformer(string performer)
{
```

```
    this.performer = performer;
```

```
public void SetLength(int length)
{
```

```
    this.length = length;
```

פעולות
קובעות

```
public void IncreaseLength(int sec)
{
```

```
    length = length + sec;
```

הפעולה מגדילה את אורך
השיר ב- sec שניות

פעולות
חישוביות

```
public void DecreaseLength(int sec)
{
```

```
    length = length - sec;
```

הפעולה מקטינה את
אורך השיר ב- sec שניות

```
public string Category()
```

```
{  
    if (length < 2*60)  
        return "short";  
    else  
        if (length > 4*60)  
            return "long";  
        else  
            return "average";  
}
```

הפעולה מחזירה מחרוזת שהיא הקטגוריה של אורך השיר אם אורכו עד 2 דקות – קצר, אם אורכו מעל 4 דקות – ארוך. אחרת – ממוצע

פעולות
חישוביות

```
public void DisplaySongDetails()
```

```
{  
    Console.WriteLine("The song name is: " + name);  
    Console.WriteLine("The performer is: " + performer);  
    Console.WriteLine("The song length in seconds is: " + length);  
}
```

הפעולה מדפיסה את תכונות השיר

מ'אול המחלקה הראשית

הפעולה הראשית במחלקה הראשית SongProgram מממשת את האלגוריתם הבא :

- 1) בנה עצם חדש ע"י בנה-שיר ("אדון עולם", "עוזי חיטמן", 190) והשם אותו במשתנה song1
- 2) בנה עצם חדש ע"י בנה-שיר () והשם אותו במשתנה song2
- 3) קבע את שם השיר song2 להיות "ימי בנימינה"
- 4) קבע את מבצע השיר song2 להיות "אהוד מנור"
- 5) קבע את אורך השיר song2 להיות 250
- 6) על העצם song1 הפעל את הפעולה הגדל-אורך-שיר-ב (20)
- 7) על העצם song1 הפעל את הפעולה הדפס-פרטי-שיר ()
- 8) הדפס את קטגוריית אורך השיר של השיר song1
- 9) הדפס את שם השיר הארוך יותר מבין השירים song1, song2, אם אורכם זהה הדפס הודעה מתאימה.



דוגמה
פתורה

```
using System;
```

```
public class SongProgram
```

```
{
```

```
// הפעולה הראשית
```

```
public static void Main(string[] args)
```

```
{
```

```
    Song song1 = new Song("אלוהים שלי", "עוזי חיטמן", 190);
```

```
    Song song2 = new Song();
```

```
    song2.SetName("ימי בנימינה");
```

```
    song2.SetPerformer("אהוד מנור");
```

```
    song2.SetLength(250);
```

```
    song1.IncreaseLength(20);
```

```
    song1.DisplaySongDetails();
```

```
    Console.WriteLine("The song category is: " + song1.Category() );
```

```
    if ( song1.GetLength() > song2.GetLength() )
```

```
        Console.WriteLine("The longest song is: " + song1.GetName());
```



```

else
if ( song1.GetLength() < song2.GetLength() )
    Console.WriteLine("The longest song is: " + song2.GetName());
else
    Console.WriteLine("The length of the songs is equal");
}
}

```

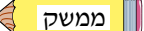
במקום הגדרת הפעולה הבונה ברירת מחדל { } Song(), וזימון פעולות קובעות, ניתן גם להגדיר פעולה בונה ללא פרמטרים אשר קולטת ערכים לתכונות:

```

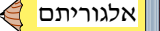
public Song()
{
    Console.WriteLine("Enter song name: ");
    name = Console.ReadLine();
    Console.WriteLine("Enter performer: ");
    performer = Console.ReadLine();
    Console.WriteLine("Enter length: ");
    length = int.Parse(Console.ReadLine());
}

```

תרגילים לחלק 3 ולסיכום הסדר

★ **תרגיל 10: כתוב ממשק עברי**  ממשק

כתוב ממשק עברי מלא לטיפוס Song

★★ **תרגיל 11: כתוב אלגוריתם**  אלגוריתם

כתוב אלגוריתם מילולי עבור הפעולה הראשית במחלקה SongProgram

★★ **תרגיל 12: הרחבת המחלקה המייצגת את הטיפוס תלמיד**  מימוש

הוסף למחלקה Student את הפעולות הבאות:



תיעוד הפעולה	הפעולה
הפעולה מקבלת מקצוע ואחוז, ומעלה את הציון במקצוע זה באחוז הנדרש.	תן-בונוס (_מקצוע, _אחוז)
הפעולה מחזירה מחרוזת המתארת את ממוצע הציונים של תלמיד: "מצטיין" – ממוצע ציוניו מ-90 ומעלה, "טוב" – ממוצע ציוניו בין 75-89, וכולל "עובר" – ממוצע ציוניו בין 55-74, וכולל "נכשלי" – ממוצע ציוניו נמוך מ-55.	התאם-הערה ()

★ ★ תרגיל 13: פיתוח מחלקה ראשית המשתמשת בטיפוס תלמיד

- הגדר מחלקה ראשית MainStudent ובה פעולה ראשית המבצעת את הדברים הבאים:
- בונה עצמים עבור שלושה תלמידים, וקובעת את הציונים שלהם. השתמש בפעולות בונות שונות ובאפשרויות עדכון שונות.
 - מדפיסה לכל תלמיד תעודה.
 - מדפיסה את שם התלמיד בעל הממוצע הגבוה ביותר.

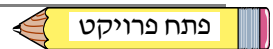
★ ★ ★ תרגיל 14: שינוי המחלקה המייצגת את הטיפוס תלמיד

- שנה את התכונות בטיפוס תלמיד כך שבמקום לייצג ציונים רק בשלושה מקצועות, יהיה מערך של 7 ציונים. בצע את כל השינויים הנדרשים בכל הפעולות בטיפוס לצורך התאמה לייצוג החדש של התכונות. התייחס לכך שמערך הציונים הוא בסדר אחיד לכל תלמיד לפי סדר נתון של המקצועות. הנח בשלב זה שלכל מקצוע יש מספר סידורי. הוסף פעולה בונה אשר מאפשרת לקלוט את הערכים של התכונות עבור העצם הנוצר. שים לב: הפעולות המאחזרות והקובעות יקבלו פרמטר נוסף שהוא המצוין של המקצוע.

★ ★ ★ תרגיל 15: שינוי המחלקה הראשית המשתמשת בטיפוס תלמיד מתרגיל 6

- הגדר מחלקה ראשית MainStudent ובה פעולה ראשית המבצעת את הסעיפים הבאים:
- בונה עצמים עבור שלושה תלמידים, וקובעת את הציונים שלהם. השתמש גם בפעולה הבונה המאפשרת לקלוט נתונים.
 - מגדירה פעולה (סטטית) המקבלת מספר מקצוע, ואת 3 התלמידים, ומחזירה את ממוצע הציונים במקצוע.
 - מגדירה פעולה (סטטית) המקבלת את 3 התלמידים וקובעת את ציוניהם לפי הכלל הזה: אם ממוצע הציונים במקצוע נמוך מ-70, יקבל כל תלמיד בונוס שהאחוז שלו הוא על פי ההפרש בין הציון הממוצע במקצוע ל-70.

כ'תוח כר'וקט'מ



בעבור כל אחד מן התרגילים 16-18 הבאים פתח פרויקט על פי השלבים הבאים:

- א. תאר את הטיפוס הנדרש על ידי דיאגרמה לתיאור טיפוס.
 - התכונות של כל טיפוס צריכות לכלול לפחות את התכונות שיפורטו בתרגיל.
 - הפעולות של כל טיפוס צריכות לכלול לפחות את הפעולות שיפורטו בתרגיל.
- ב. כתוב שתי פעולות בונות.
- ג. כתוב פעולות קובעות ומאחזרות לכל תכונה.
- ד. הוסף פעולות כך שפרט מן הפעולות לעיל תהיינה לפחות שתי פעולות המחזירות ערך ולפחות שתי פעולות שאינן מחזירות ערך.
- ה. פתח מחלקה בשפת התכנות למימוש הטיפוס שהגדרת.
- ו. פתח מחלקה ראשית ובה יבנו לפחות שני עצמים, ויופעלו עליהם פעולות.
- הגדר פעולה סטטית אחת בתכנית הראשית שתבצע משימה כרצונך.

תרגיל 16: הטיפוס קובץ ☆☆☆

כל עבודה במערכת ממוחשבת מושתתת על עבודה עם קבצים.
המאפיינים הבסיסיים של כל קובץ הם:
תכונות: שם הקובץ, סוג הקובץ, גודל הקובץ, תאריך יצירת הקובץ, האם פתוח?
פעולות: קובץ חדש, פתח קובץ, שמור קובץ, סגור קובץ, עדכן קובץ, שנה שם.
על התכנית הראשית לכלול בנוסף לדברים שפורטו לעיל גם קטע תכנית אשר ידפיס כמה מן הקבצים שנבנו בתכנית עם סיומת doc.

תרגיל 17: הטיפוסים חשבון עובר ושב וקופת גמל ☆☆☆☆

במערכת הבנקאית מנוהלים סוגים שונים של חשבונות. בפרויקט זה נתייחס לשני סוגים של חשבונות, חשבון עובר ושב וחשבון קופת גמל.
המאפיינים הבסיסיים של חשבון עובר ושב הם:
תכונות: מספר בנק, מספר סניף, מספר חשבון, ת.ז. של הבעלים של החשבון, יתרה, גובה משיכת יתר מותרת.
פעולות חישוביות: הפקדת סכום כסף, משיכת סכום כסף אם מותר.
המאפיינים הבסיסיים של קופת גמל הם:
תכונות: סוג קופת גמל, מספר חשבון, ת.ז. של הבעלים של החשבון, שנת פתיחה, גובה הפקדה חודשי, יתרה נוכחית.
פעולות חישוביות: בצע הפקדה חודשית, מספר שנים שנותרו (עד 15 שנה) עד לפדיון הקופה.
על התכנית הראשית לבנות לפחות עצם אחד מכל טיפוס לאותו אדם. יש לבצע משיכה מחשבון עובר ושב בגובה הנדרש להפקדה בקופת הגמל, ולהפקיד אותו לקופה. אם אין סכום מספיק יש להדפיס הודעה "קופת הגמל בסכנת סגירה".

תרגיל 18: הטיפוס רובוט ☆☆☆☆

על לוח משחק ובו משבצות עוקבות המסומנות במספרים מ-1 עד 100, נעים רובוטים. הרובוטים יכולים לנוע קדימה או אחורה. רובוט אשר חורג בצעדיו מגבולות לוח המשחק יוצא מן המשחק. המאפיינים הבסיסיים של כל רובוט הם:
תכונות: צבע, מספר משבצת עליו הוא עומד, האם נמצא במשחק?
פעולות חישוביות: צעד קדימה - הרובוט נע משבצת אחת קדימה, צעד אחורה - הרובוט נע משבצת אחת לאחור, קפוץ קדימה ב- n צעדים, קפוץ אחורה ב- n צעדים.
על התכנית הראשית לבנות שני רובוטים ולנהל ביניהם משחק אקראי. כללי המשחק הם: שני הרובוטים מתחילים במשבצת מספר 10. בכל שלב עבור כל רובוט מוגרלת פעולת ההתקדמות שלו (4 אפשרויות). אם הוגרלה פעולת קפיצה יש להגריל גם את ערך הקפיצה מ-1-6. הרובוט הראשון שנעמד על המשבצת 100 הוא המנצח. אם רובוט מגיע למשבצת עליה נמצא רובוט אחר הוא מוציא את הרובוט השני מן המשחק והוא מוכרז כמנצח.

דגלים לס'כלום הסרק

- ✓ **עצם (object)** – מופע של טיפוס. לפעמים אומרים **דוגמה** או **מקרה פרטי** או **יישות (entity)**. עצם מיוצג על ידי התכונות המוגדרות בטיפוס שלו והערכים שלהן, וניתן להפעיל עליו את הפעולות המוגדרות בטיפוס שלו.
- מצב העצם (object state)** – מכלול ערכי התכונות של עצם ברגע מסוים.
- זיהוי עצם** – עצמים נבדלים זה מזה על פי המזהה שלהם. לכל עצם שנבנה יש מזהה יחודי. מזהה בשפת התכנות הוא משתנה המכיל הפנייה אל העצם.
- ✓ **טיפוס (type)** – הגדרה של המאפיינים המשותפים ל קבוצת עצמים. המאפיינים הם תכונות ופעולות. טיפוס הוא **תבנית** שממנה ניתן לבנות עצמים. ניתן לבנות עצמים רבים מאותו טיפוס.
- ✓ **מחלקה (class)** – קוד המממש הגדרה של טיפוס.
- ✓ **בנייה של עצם (object creation)** – תהליך הבונה עצם חדש על פי תבנית הטיפוס שלו. לעצם שנבנה יש את התכונות שהוגדרו בטיפוס שלו והן מקבלות ערכים בתהליך הבניה. רק אחרי שעצם נבנה ניתן להפעיל עליו את הפעולות המוגדרות בטיפוס שלו. בנייה של עצם נעשית על ידי פעולה מיוחדת הנקראת **פעולה בונה (constructor)**.
- ✓ **תכונה (attribute)** – מאפיין המוגדר בטיפוס. בטיפוס מוגדרות מספר תכונות כנדרש לתאור היישויות. לכל עצם מן הטיפוס יש את כל התכונות המוגדרות בטיפוס שלו.
- התכונות מוגדרות בגוף המחלקה (ולא בגוף פעולה). תכונות מוגדרות עם בקרת גישה `private`, כדי לא לאפשר גישה ממחלקות אחרות. גישה לתכונות ממחלקות אחרות תתבצע בעזרת פעולות מאחזרות.
- ערך תכונה (attribute value)** – הערך המושם בתכונה. עבור כל עצם לכל תכונה שלו יש ערך המתאר את המאפיין בעצם מסוים זה.
- ✓ **פעולה (method)** – מוגדרת בטיפוס וניתנת לביצוע על עצם מן הטיפוס. פעולות יכולות: לאחזר את ערכי תכונות העצם, לשנות את ערכי תכונות העצם או לבצע משימות אלגוריתמיות אחרות. פעולות בטיפוס מוגדרות **ללא static**, כי הן פועלות על עצמים.
- הפעלת פעולה על עצם (method invocation)** – אחרי שעצם נבנה ניתן להפעיל עליו את הפעולות המוגדרות בטיפוס שלו. פעולה מתבצעת על עצם על ידי זימון מתאים למבנה שלה. (בשונה מפעולות סטטיות (static) שלא מופעלות על עצמים).
- ✓ **מחלקה המייצגת טיפוס** – מחלקה המממשת טיפוס כוללת: תכונות, פעולות בונות, פעולות קובעות, פעולות מאחזרות ופעולות חישוביות.
 - **פעולה בונה** – פעולה מיוחדת הבונה ומחזירה עצם מן הטיפוס.
 - **פעולה קובעת** – פעולה הקובעת ערך של תכונה. לכל תכונה מוגדרת פעולה קובעת. מקובל ששם הפעולה מתחיל ב `Set`.
 - **פעולה מאחזרת** – פעולה המאחזרת ערך של תכונה. לכל תכונה מוגדרת פעולה מאחזרת. מקובל ששם הפעולה מתחיל ב `Get`.
- ✓ **מחלקה ראשית** – מבצעת אלגוריתם ראשי, בה מוגדרים עצמים ומופעלות עליהם פעולות.