פרק 5 – ביצוע מותנה

בשני הפרקים הקודמים ראינו אלגוריתמים שבמהלך ביצועם מתבצעת כל אחת מהוראות האלגוריתם. בפרק זה נכיר אלגוריתמים אשר במהלך ביצועם לא מתבצעות תמיד כל הוראות האלגוריתם. אלגוריתמים אלה כוללים הוראות המורות על ביצוע קבוצת הוראות אחת או על קבוצת הוראות אחרת, בהתאם לקיומו או לאי-קיומו של תנאי. הוראות אלו נקראות **הוראות לביצוע-בתנאי**. קיומו של התנאי תלוי בקלט לאלגוריתם.

למשל כאשר נערכות בחירות בין שני מועמדים, משווים את מספרי הקולות לכל מועמד. המועמד שצבר יותר קולות הוא המנצח בבחירות. הקלט של אלגוריתם להכרזת המנצח יהיה מספר הקולות אשר צבר כל מועמד. אם יתקיים התנאי שהנתון הראשון בקלט גדול מן השני אז תתבצע באלגוריתם הוראה להכרזת המועמד הראשון כמנצח. אחרת תתבצע הוראה להכרזת המועמד השני כמנצח.

5.1 הוראה לביצוע-בתנאי

הוראה לביצוע-בתנאי במבנה *אם... אחרת...*

בעיה 1

מטרת הבעיה ופתרונה: הצגת אלגוריתם הכולל הוראה לביצוע-בתנאי.

> פָּלִינְדְּרוֹם (palindrome) הוא מילה, מספר או משפט שניתן לקרוא משני הכיוונים, משמאל לימין ומימין לשמאל, ולקבל אותה תוצאה. למשל השם ישי הוא שם פלינדרומי, וכן המילים זוז, שמש, הסוסה. המילה הפלינדרומית הארוכה ביותר בעברית שיש לה משמעות היא "ולכשתשכלו". המשפט: "ילד כותב בתוך דלי" גם הוא פלינדרומי. המספרים 17371 ו-4994 הם דוגמאות למספרים פלינדרומים.
> פתחו אלגוריתם אשר הקלט שלו הוא מספר שלם חיובי תלת-ספרתי, והפלט שלו הוא הודעה אם המספר הנתון הוא פלינדרום. ישמו את האלגוריתם בתוכנית מחשב בשפת C#.

ניתוח הבעיה בעזרת דוגמאות

דוגמאות למספרים שלמים חיוביים תלת-ספרתיים פלינדרומים: 787 424 777

דוגמאות למספרים שלמים חיוביים תלת-ספרתיים שאינם פלינדרומים: 778 234 192

שאלה 5.1

הגדירו כלל פשוט המתאר מתי מספר הוא פלינדרום ומתי אינו פלינדרום.

בדוגמאות הפשוטות שבחנו התברר שמספר הוא פלינדרומי רק כאשר ספרת האחדות שווה לספרת המאות.

פירוק הבעיה לתת-משימות

על פי ניתוח הבעיה נפרק את המשימה העומדת לפנינו באופן הבא:

1. קליטת מספר שלם חיובי תלת-ספרתי.
2. מציאת ספרת האחדות.

3. מציאת ספרת המאות.
4. השוואת הספרות. אם הספרות שוות נציג הודעה שהמספר פלינדרומי, אחרת נציג הודעה שהמספר אינו פלינדרומי.

בחירת משתנים

num – שלם, ישמור את המספר התלת-ספרתי הנקלט.
units – שלם, ישמור את ספרת האחדות.
hundreds - שלם, ישמור את ספרת המאות.

האלגוריתם

את תת-משימה 2 ו-3 למדנו לבצע בפרק הקודם. על מנת למצוא את ספרת האחדות של מספר כלשהו, נחשב את תוצאת שארית החילוק של המספר ב-10. על מנת למצוא את ספרת המאות של מספר תלת-ספרתי, נחשב את תוצאת החילוק של המספר ב-100.

כיצד ננסח באלגוריתם את תת-משימה 4?
ננסח תנאי אשר על פי קיומו ניתן לקבוע אם המספר הוא פלינדרומי. התנאי יהיה: ספרת האחדות שווה לספרת המאות.
אם התנאי מתקיים יש להודיע: המספר פלינדרומי.
אם התנאי לא מתקיים יש להודיע: המספר אינו פלינדרומי.

הוראה זו מבטאת את הרעיון של בחירה באחת מבין שתי אפשרויות לביצוע על פי תנאי. נציג אותה במבנה הבא:

אם ספרת האחדות שווה לספרת המאות
הצג כפלט: המספר פלינדרומי
אחרת
הצג כפלט: המספר אינו פלינדרומי

הוראה במבנה זה נקראת **הוראה לביצוע-בתנאי**. הוראה לביצוע-בתנאי היא הוראת **בקרה**, משום שהיא משפיעה על מהלך הביצוע של האלגוריתם, כלומר קובעת אם יבוצעו הוראות אלו או אחרות.

יישום האלגוריתם

ההוראה לביצוע-בתנאי המופיעה באלגוריתם מיושמת בשפת C# בהוראת **if…else…**, ואופן כתיבתה דומה לצורת הכתיבה העברית *אם...אחרת...*

אבל כיצד כותבים ב-C# תנאי כמו זה המופיע באלגוריתם: *ספרת האחדות שווה לספרת המאות*? פעולת ההשוואה נכתבת בעזרת סימן הפעולה == (שני סימני שוויון רצופים).

לכן, את תת-משימה 4 באלגוריתם ניתן ליישם בשפת C# כך:

```
if (units == hundreds)
{
    Console.WriteLine("The number {0} is a palindrome", num);
}
else
{
    Console.WriteLine("The number {0} is not a palindrome", num);
}
```

התוכנית המלאה

```
/*
הקלט: מספר שלם חיובי תלת-ספרתי
הפלט: הודעה אם המספר הוא פלינדרום
*/
using System;
public class Palindrome
{
    public static void Main ()
    {
        // הצהרה על משתנים בתוכנית
        int num;         // מספר שלם חיובי תלת-ספרתי
        int units;       // ספרת האחדות
        int hundreds;    // ספרת המאות
        // קליטת המשתנים
1.      Console.Write("Enter a 3 digit number: ");
2.      num = int.Parse(Console.ReadLine());
        // פירוק ספרת האחדות וספרת המאות
3.      units = num % 10;
4.      hundreds = num / 100;
        //  ההוראה לביצוע-בתנאי
5.      if (units == hundreds)
        {
            //הצגת הודעה: המספר פלינדרומי
5.1.        Console.WriteLine("{0} is a palindrome", num);
        }
6.      else
        {
            //הצגת הודעה: המספר אינו פלינדרומי
6.1.        Console.WriteLine("{0} is not a palindrome", num);
        }
    } // Main
} // class Palindrome
```

המעקב

נבדוק את התוכנית Palindrome באמצעות מעקב אחר ביצועה עבור דוגמאות קלט שונות.

עבור הקלט 363, יתקיים התנאי *ספרת האחדות שווה לספרת המאות* ותתקבל טבלת המעקב הבאה:

מספר השורה	**המשפט לביצוע**	num	units	hundreds	units == hundreds	**פלט**
1	Console.Write("Enter a 3 digit number: ");	?	?	?		Enter a 3 digit number:
2	num = **int**.Parse(...);	363	?	?		
3	units = num % 10;	363	3	?		
4	hundreds = num / 100;	363	3	3		
5	**if** (units == hundreds)	363	3	3	אמת	
5.1	Console.WriteLine("{0} is a palindrome", num);	363	3	3		363 is a palindrome

עבור הקלט 366, לא מתקיים התנאי, ומתקבלת טבלת המעקב הבאה:

מספר השורה	המשפט לביצוע	num	units	hundreds	units == hundreds	פלט
1	`Console.Write("Enter a 3 digit number: ");`	?	?	?		Enter a 3 digit number:
2	`num = int.Parse(...);`	366	?	?		
3	`units = num % 10;`	366	6	?		
4	`hundreds = num / 100;`	366	6	3		
5	`if (units == hundreds)`	366	6	3	שקר	
6.1	`Console.Write("{0} is not a palindrome",num);`	366	6	3		366 is not a palindrome

שימו ♥ להבדל בין עמודת "המשפט לביצוע" בשתי הטבלאות. בטבלה הראשונה מופיע המשפט:

```
Console.WriteLine("{0} is a palindrome", num);
```

זאת מכיוון שהתנאי מתקיים ולכן מתבצע תחום ה-**if** של משפט ה-**if**.

לעומת זאת, בטבלה השנייה מופיע המשפט:

```
Console.WriteLine("{0} is not a palindrome", num);
```

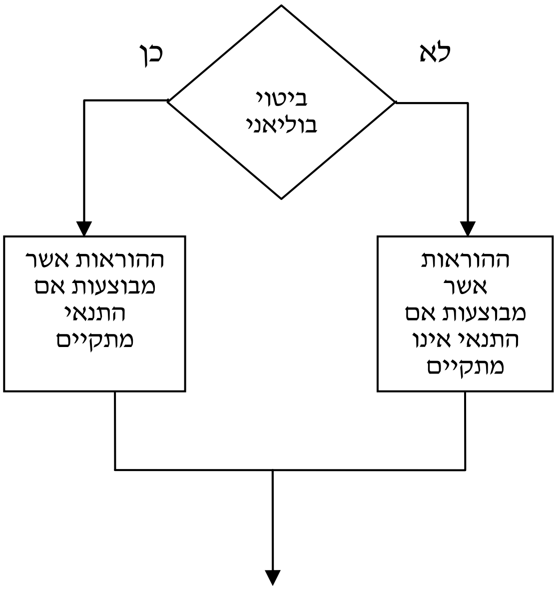
זאת מכיוון שהתנאי אינו מתקיים ולכן מתבצע תחום ה-**else** של המשפט.

סוף פתרון בעיה 1

הוראה לביצוע-בתנאי כוללת בתוכה כמובן תנאי, שמכוון את המשך הביצוע. באלגוריתם שנתנו לפתרון בעיה 1 התנאי ששילבנו בהוראה הוא *ספרת האחדות שווה לספרת המאות*.

התנאי העומד בבסיסה של הוראת ביצוע-בתנאי מיוצג ב**ביטוי בוליאני**. כמו ביטוי חשבוני, גם לביטוי בוליאני יש ערך. אלא שערך זה אינו ערך מספרי. ערכו של ביטוי בוליאני יכול להיות אחד משניים – אמת (**true**) או שקר (**false**). אם התנאי שמייצג הביטוי הבוליאני מתקיים אז ערכו של הביטוי הבוליאני הוא **true**. אם התנאי שמייצג הביטוי הבוליאני אינו מתקיים אז ערכו של הביטוי הבוליאני הוא **false**.

ניתן להמחיש את המשמעות של הוראה לביצוע-בתנאי באמצעות תרשים הזרימה הבא:



הידעת?
המילה "בוליאני" נגזרה משמו של המתמטיקאי האנגלי ג'ורג' בול (1864-1815). הוא היה בין שאר עיסוקיו לוגיקן – חוקר תורת ההיגיון. הוא היה המתמטיקאי הראשון שראה בלוגיקה ענף של המתמטיקה. הוא נחשב לאבי הלוגיקה הסימבולית (אלגברה של הלוגיקה) הקרויה היום אלגברה בוליאנית.

במהלך הספר לא נציג אלגוריתמים באמצעות תרשימים. הצגה זו ארוכה מאוד וכן שונה מאוד מן המראה של תוכנית מחשב. אם השימוש בתרשימים מסייע לכם, כדאי לכם להשתמש בהם מדי פעם במהלך פיתוח אלגוריתם.

לעתים ננסח את התנאי באופן מילולי ולעתים נעדיף לנסח אותו בעזרת סימנים. ניתן לבחור בכל צורת ניסוח, כל עוד התנאי המתקבל הוא ברור וחד-משמעי. למשל עבור המשימה: השוואת ערכי המשתנים a ו-b והשמת הערך הקטן מביניהם במשתנה min, ניתן לנסח את התנאי במילים או בעזרת סמלים:

אם ערכו של a *קטן מערכו של* b　　　או　　　*אם* $a < b$

נכיר ביטויים בוליאניים המציינים השוואה של ערכים, של משתנים ושל ביטויים חשבוניים. למשל: units = = hundreds , $a + b > 0$ ו-$a < b$.

לסיכום – נציג את אופן הכתיבה האלגוריתמית של הוראה לביצוע-בתנאי ואת יישומה בשפת C#:

הוראה לביצוע-בתנאי נכתבת בצורה זו:

אם <ביטוי בוליאני>
　　<סדרת הוראות 1>
אחרת
　　<סדרת הוראות 2>

ביטוי בוליאני מייצג תנאי, שערכו יכול להיות **true** (אמיתי) או **false** (שקרי).
ביצוע של הוראה לביצוע-בתנאי מתחיל תמיד בחישוב ערכו של הביטוי הבוליאני. אם ערכו אמיתי, תתבצע סדרת ההוראות הראשונה, אחרת תתבצע סדרת ההוראות השנייה.

בשפת C# הוראה לביצוע-בתנאי מיושמת במשפט `if`.

מבנה משפט `if` ב-C# הוא:

```
if (ביטוי בוליאני)
{
    ההוראות אשר יבוצעו אם התנאי מתקיים
}
else
{
    ההוראות אשר יבוצעו אם התנאי אינו מתקיים
}
```

החלק שנמצא בתוך זוג הסוגריים המסולסלים ({...}) הראשון נקרא **תחום ה-`if`** ובו נמצאות ההוראות אשר יבוצעו אם התנאי מתקיים. החלק שנמצא בתוך זוג הסוגריים המסולסלים השני נקרא **תחום ה-`else`** ובו נמצאות ההוראות אשר מבוצעות אם התנאי אינו מתקיים.

שימו ♥: במקרים שיש הוראה אחת לביצוע בתחום ה-**if**, אפשר להשמיט את הסוגריים המסולסלים של התחום. גם במקרים שיש הוראה אחת לביצוע בתחום ה-**else**, דין הסוגריים זהה וניתן להשמיטם.

הביטוי הבוליאני שהופיע במשפט ה-`if` שראינו בפתרון בעיה 1 השתמש בסימן ההשוואה `==`. בשפת C# קיימים סימני השוואה נוספים. בטבלה הבאה מובאים סימני ההשוואה בשפת C#.

סימן ההשוואה ב-C#	**סימן ההשוואה המקובל במתמטיקה**	**משמעות סימן ההשוואה**	**דוגמה ב-C#**	**דוגמה במתמטיקה**
==	=	שווה	`x == 5`	$x = 5$
!=	≠	שונה	`x != y`	$x \neq y$
<	<	קטן	`x < 2`	$x < 2$
<=	≤	קטן או שווה	`x <= 1`	$x \leq 2$
>	>	גדול	`y > 0`	$y > 0$
>=	≥	גדול או שווה	`y >= 8`	$y \geq 8$

סימני ההשוואה בשפת C#

שימו ♥ : הסימנים המתמטיים ≤, ≥, = ו - ≠ כתובים ב-C# בצורה שונה במקצת. בשפת C# ההשוואה שבתוך התנאי מתבצעת על ידי סימן הפעולה `==` ולא על ידי הסימן `=`. הסימן `=` שמור ב-C# להשמה.

שאלה 5.2

נסחו הוראה לביצוע-בתנאי עבור כל אחת מן המשימות הבאות:

א. השוואת ערכי המשתנים `a` ו-`b` והצגת הודעה אם הערכים שווים או שונים.

ב. השוואת ערכי המשתנים `a` ו-`b` והפחתת ערכו של המשתנה הקטן מהמשתנה הגדול. אם שני המשתנים שווים יש להפחית את ערכו של b מערכו של a.

שאלה 5.3

כתבו כל אחד מן התנאים המילוליים הבאים כביטוי בוליאני המשתמש בסימני ההשוואה של C#:

תנאי	**ביטוי בוליאני**
ערך המשתנה `a` שווה ל-0.	
ערך המשתנה `a` שווה לערך המשתנה `b`.	
ערך המשתנה `a` שווה לכפליים ערכו של המשתנה `b`.	
ערך המשתנה `a` שונה מערך המשתנה b.	
סכום ערכי המשתנים `a` ו-`b` קטן או שווה ל-10.	

שאלה 5.4

נניח שערכי המשתנים `a` ו-`b` הם 1 ו-2 בהתאמה. ציינו עבור כל אחד מן הביטויים הבוליאניים הבאים אם ערכו **true** או **false**.

ביטוי בוליאני	**ערך**
`a == 1`	
`a == b`	
`3a == b + 1`	
`2a <= b`	
`2a != b`	

שאלה 5.5

במשתנים `girafaHeight` ו-`girafHeight` שמורים הגבהים של ג'ירף ושל ג'ירפה. כתבו משפט **if** מתאים לביצוע המשימות הבאות:

א. הצגת הודעה אם הג'ירפה גבוהה מ-1.70 מ', או לא.

ב. הצגת הודעה אם הג'ירף גבוה יותר או אינו גבוה יותר מהג'ירפה.

כפי שראינו בפתרון בעיה 1: בטבלת מעקב אחר מהלך ביצוע של תוכנית הכוללת משפט **if**, נוח להוסיף עמודה עבור הביטוי הבוליאני, ולציין בה את ערך הביטוי. בטבלה שבפתרון בעיה 1 כתבנו "אמת" או "שקר". מעתה נכתוב **true** או **false**.

שימו ♥: כאשר מופיע בתוכנית משפט **if** (הוראה לביצוע-בתנאי), יש לבדוק את מהלך הביצוע עבור דוגמאות קלט מגוונות. על הדוגמאות לכלול קלט שיביא לכך שערכו של הביטוי הבוליאני שבמשפט יהיה **true** וכן קלט שיביא לכך שערכו של הביטוי הבוליאני שבמשפט יהיה **false**. קלטים כאלו נקראים קלטים מייצגים.

לא מספיק לבדוק את מהלך הביצוע רק עבור אחד משני המקרים, כיוון שמקרה אחד אינו מעיד על האחר. עלינו לבדוק את שניהם, כפי שמדגימה השאלה הבאה.

שאלה 5.6

מטרתו של משפט ה-**if** הבא היא השמת הערך הגדול מבין המשתנים `a` ו-`b` במשתנה `max`. חישוב מקסימום היא תבנית שימושית מאוד, שמשמשת בפתרון בעיות רבות.

```
if (a > b)
{
      max = a;
}
else
{
      max = b;
}
```

בחרו שתי דוגמאות של קלטים מייצגים לבדיקת המשפט.

באחת יהיה ערכו ההתחלתי של `a` גדול מערכו ההתחלתי של `b`, ובשנייה יהיה ערכו ההתחלתי של `a` קטן מערכו ההתחלתי של `b`.

בדקו את מהלך ביצוע המשפט באמצעות טבלת מעקב עבור כל אחת מן הדוגמאות.

מה יהיה מהלך ביצוע המשפט עבור המקרה שבו הערכים ההתחלתיים של `a` ו-`b` שווים?

הוראה לביצוע-בתנאי במבנה *אם...*

לעתים ברצוננו לבצע חלק של אלגוריתם כאשר תנאי מסוים מתקיים, ואיננו רוצים לבצע מאומה כאשר התנאי אינו מתקיים. נראה זאת בפתרון הבעיה הבאה.

בעיה 2

מטרת הבעיה ופתרונה: הצגת הוראה לביצוע-בתנאי במבנה *אם...*(ללא החלק *אחרת...*).

> תלמידי כיתות י' בבית הספר טסים לירת. יש להזמין מספר מתאים של חלליות כך שלכל תלמיד יהיה מקום ישיבה ושמספר החלליות יהיה קטן ככל האפשר.

פתחו וישמו אלגוריתם להזמנת חלליות לירח כך שהקלט שלו הוא מספר התלמידים ומספר המושבים בחללית, והפלט שלו הוא מספר החלליות שיש להזמין.

ניתוח הבעיה בעזרת דוגמאות

ייתכן כי מספר התלמידים הוא כפולה של מספר המושבים בחללית. במקרה כזה בכל החלליות שיוזמנו יהיו כל המושבים תפוסים.

לעומת זאת, ייתכן כי מספר התלמידים אינו כפולה של מספר המושבים בחללית. במקרה כזה תוזמן גם חללית אחת אשר רק חלק מהמושבים בה יהיו תפוסים .

שאלה 5.7

בחרו שתי דוגמאות קלט מייצגות וציינו את הפלט עבור כל אחת מהן.

? כיצד נחשב את מספר החלליות הדרוש?

יש לחשב את המנה ואת השארית של חלוקת מספר התלמידים במספר המושבים בחללית. מנת החלוקה שווה למספר החלליות המלאות. שארית החלוקה תקבע אם יש צורך בחללית נוספת. אם שארית החלוקה היא 0 הרי מספר התלמידים הוא כפולה של מספר המושבים בחללית. אחרת יש להזמין חללית נוספת שתפוסתה תהיה חלקית ושווה לשארית.

פירוק הבעיה לתת-משימות

1. קליטת מספר התלמידים ומספר המושבים
2. חישוב מספר החלליות המלאות
3. חישוב מספר התלמידים שיישארו לאחר מילוי החלליות המלאות
4. אם צריך חללית נוספת, חישוב מספר החלליות הכולל
5. הצגת הפלט: מספר החלליות

בחירת משתנים

נבחר את המשתנים הבאים מטיפוס שלם:

studentsNum – ישמור את מספר התלמידים.

seatsPerShip – ישמור את מספר המושבים בחללית.

shipsNum – ישמור את מספר החלליות שיש להזמין.

leftoverNum – ישמור את מספר התלמידים אשר ייוותרו לאחר מילוי החלליות המלאות.

האלגוריתם

כיצד ננסח באלגוריתם את תת-משימה 4?

ניתן לתאר תת-משימה זו באופן הבא:

אם מספר התלמידים שנשארו גדול מ-0

הגדל ב-1 את shipsNum

האלגוריתם לפתרון הבעיה כולל הוראה לביצוע-בתנאי במבנה *אם...* (ללא החלק *אחרת...*):

> המשמעות של **הוראה לביצוע-בתנאי במבנה** *אם...* היא שאם התנאי שבהוראה מתקיים יבוצעו ההוראות המתאימות, אחרת לא יבוצע דבר.

יישום האלגוריתם

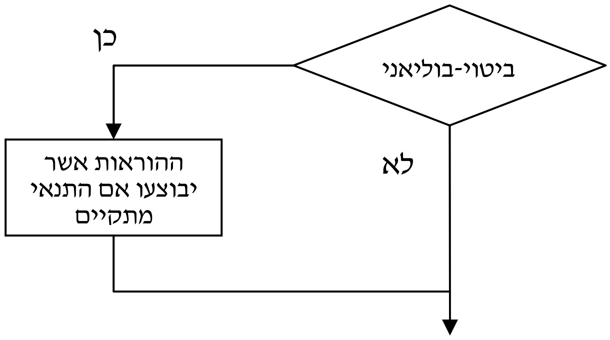
```
/*
קלט: מספר התלמידים והמושבים
 פלט: מספר החלליות להזמנה
*/
using System;
public class ShipspaceOrder
{
    public static void Main ()
    {
        // הצהרה על משתנים בתוכנית
        int studentsNum;    // מספר התלמידים
        int seatsPerShip;  // מספר המושבים בחללית
        int shipsNum;        // מספר החלליות שיש להזמין
        int leftoverNum;    // מספר התלמידים שיישארו
            // לאחר מילוי החלליות המלאות
        // קליטת המשתנים
1.      Console.Write("Enter number of students:");
2.      studentsNum = int.Parse(Console.ReadLine());
3.      Console.Write("Enter number of seats in a spaceship: ");
4.      seatsPerShip = int.Parse(Console.ReadLine());
              // חישוב מספר החלליות המלאות
5.      shipsNum = studentsNum / seatsPerShip;
        // חישוב מספר התלמידים שיוותרו
6.      leftoverNum = studentsNum % seatsPerShip;
        //   ההוראה לביצוע-בתנאי
7.      if (leftoverNum > 0) //.דרושה חללית נוספת שתפוסתה תהיה חלקית
        {
7.1.        shipsNum = shipsNum + 1;
        }
                Console.WriteLine("The number of spaceships is {0}",
                                                        shipsNum);
    } // Main
} // class ShipspaceOrder
```

שאלה 5.8

עקבו אחר ביצוע התוכנית באמצעות שתי טבלאות מעקב. האחת, כאשר התנאי מתקיים, והשנייה, כאשר אין התנאי מתקיים. לדוגמה: עבור הקלט 120 תלמידים ו-40 מושבים בכל חללית התנאי לא יתקיים, ואילו עבור הקלט 120 תלמידים ו-50 מושבים בכל חללית התנאי יתקיים.

סוף פתרון בעיה 2

ניתן להמחיש הוראה לביצוע-בתנאי במבנה *אם*... באמצעות תרשים הזרימה הבא:



בשפת C# מבנה משפט ה-`if` ליישום הוראה לביצוע-בתנאי במבנה *אם*... הוא:

```
if (ביטוי בוליאני)
{
      ההוראות אשר יבוצעו אם התנאי מתקיים
}
```

שימו ♥: במקרים שיש הוראה אחת לביצוע, אפשר להשמיט את הסוגריים המסולסלים של תחום ה-**if**.

שימו ♥: הערה שמלווה את התנאי (דרושה חללית...) מבהירה ביטוי בוליאני שמשמעותו איננה ברורה מיד עם קריאתו. נקרא להערה זו **תיאור משמעות של קיום תנאי**.

תיאור משמעות של קיום תנאי הוא תיעוד המסביר את תפקידו של ביטוי בוליאני בהוראה לביצוע-בתנאי. תיאור משמעות קיום תנאי מסייע לנו בקריאת תוכנית ובהבנתה.

בתוכניות שנפתח נשתדל לצרף תיאורי משמעות של קיום או של אי-קיום תנאי במשפטי `if` אשר כדאי להבהירם.

שאלה 5.9

בנו טבלת מעקב אחר ביצוע התוכנית `SpaceshipOrder` לפתרון בעיה 2 עבור הקלט: 180 תלמידים ו-60 מושבים.

שאלה 5.10

ביטוי בוליאני עשוי לכלול השוואה בין ביטויים חשבוניים שאינם פשוטים. למשל, `(a / 2) == 0` או `(a + b) == (c + d)`.

כתבו משפט `if` אשר בו:

א. אם ערכו של `a` גדול מפעמיים ערכו של b, מוכפל ערכו של `c` ב-2.

ב. אם ערכו של `c` קטן מסכום ערכי `a` ו-b, מופחת מ-`c` ערכו של `a`.

ג. אם ערכו של `a` הוא כפולה של 10, מושם ב-`b` ערכו של `a`.

ד. אם מכפלת ערכי `a` ו-`b` גדולה מסכום ערכי `b` ו-`c`, סימנו של הערך הנתון ב-`c` מתהפך.

שאלה 5.11

המשתנה a מכיל מספר שלם חיובי קטן מ-100.

השלימו את תיאור המשמעות של קיום תנאי בכל אחד מן המשפטים הבאים, כלומר הסבירו את תפקידו של כל תנאי ומה הוא בודק:

א.

```
//___________________________
if ((a / 10) > 5)
    Console.WriteLine("Pass");
```

ב.

```
//___________________________
if (a == (a % 10))
    Console.WriteLine("A");
```

שאלה 5.12

נתון קטע התוכנית הבא, שהמשתנים בו הם מטיפוס שלם:

```
max = a;
if (b > a)
    max = b;
```

א. בנו טבלאות מעקב אחר מהלך ביצוע קטע התוכנית עבור הערכים ההתחלתיים 30 ו-30 ועבור הערכים ההתחלתיים 40 ו-70 במשתנים a ו-b בהתאמה.

ב. מהי מטרת קטע התוכנית?

ג. נתון קטע התוכנית הבא:

```
if (a >= b)
    max = a;
else
    max = b;
```

האם יש הבדל בין מטרת קטע התוכנית הנתון בסעיף זה ובין מטרת קטע התוכנית שהוצג בתחילת השאלה? הסבירו את תשובתכם.

להעמקה בתבנית ***מציאת מקסימום*** פנו לסעיף התבניות המופיע בסוף הפרק.

שאלה 5.13

פתחו אלגוריתם אשר הקלט שלו הוא שני ציונים של תלמיד, שערכם הוא מספר שלם בין 0 ל-100 והפלט שלו הוא מספר המציין כמה מן הציונים גבוהים מ-80.

א. מהם ערכי הפלט האפשריים?

ב. ישמו את האלגוריתם בשפת C#.

התניית ביצוע של שתי הוראות או יותר

עד כה ראינו הוראות פשוטות לביצוע-בתנאי: אם התנאי התקיים התבצעה הוראה יחידה. גם במקרה שהתנאי לא התקיים התבצעה הוראה יחידה. בבעיה הבאה נציג הוראה לביצוע-בתנאי שתחומיה השונים כוללים יותר מהוראה אחת, ומורים על ביצוע כמה תת-משימות.

בעיה 3

מטרת הבעיה ופתרונה: הצגת הוראה לביצוע-בתנאי שתחומיה כוללים מספר הוראות.

> פתחו אלגוריתם המתאר משחק שנקרא "שש אש". הקלט של האלגוריתם הוא מספר שלם x. אם x מתחלק ב-6 יזכה המשתתף ב- 6·x ₪, אך אם x אינו מתחלק ב-6, יפסיד המשתתף 10·x ₪. פלט האלגוריתם הוא הודעת סכום הזכייה או ההפסד מוקף בכוכביות. ישמו את האלגוריתם בשפת C#.

פירוק הבעיה לתת-משימות

1. קליטת המספר
2. חישוב סכום הזכייה או ההפסד
3. הצגת הסכום מוקף בכוכביות בצירוף הודעה מתאימה

בחירת משתנים

נבחר את המשתנים הבאים מטיפוס שלם:

num – לשמירת המספר הניתן כקלט

sum – לשמירת סכום הזכייה או ההפסד

האלגוריתם

1. *קלוט מספר ב-*num
2. *אם ערכו של* num *מתחלק ב-6 ללא שארית*
   - 2.1. *חשב את מכפלת* num *ב-6 והשם ב-*sum
   - 2.2. *הצג הודעה על זכייה בסכום* sum *המוקפת בכוכביות*
3. *אחרת*
   - 3.1. *חשב את מכפלת* num *ב-10 והשם ב-*sum
   - 3.2. *הצג הודעה על הפסד של הסכום* sum *המוקפת בכוכביות*

יישום האלגוריתם

```
/*
קלט: מספר שלם
פלט: הודעה על זכייה אם המספר מתחלק ב-6, או על הפסד אם המספר אינו
מתחלק ב-6. ההודעה תכלול את סכום הזכייה או ההפסד
*/
using System;
public class SheshEsh
{
    public static void Main ()
```

```
    {
        // הצהרה על משתנים בתוכנית
        int num;   //המספר שנקלט
        int sum;   //סכום הזכייה או ההפסד
        // קליטת המשתנים
1.      Console.Write("Enter a number: ");
2.      num = int.Parse(Console.ReadLine());
        // ההוראה לביצוע-בתנאי: חישוב סכום הזכייה או ההפסד
3.      if (num % 6 == 0)
        {   // הזכייה
3.1.        sum = num * 6;
3.2.        Console.WriteLine("**********************");
3.3.        Console.WriteLine("* You won {0} shekels *", sum);
3.4.        Console.WriteLine("**********************");
        }
4.      else
        {     // ההפסד
4.1.        sum = num * 10;
4.2.        Console.WriteLine("************************");
4.3.        Console.WriteLine("* You  lost {0} shekels *", sum);
4.4.        Console.WriteLine("************************");
        }
    }// Main
}// class SheshEsh
```

מעקב

נעקוב אחר מהלך ביצוע התוכנית SheshEsh עבור הקלט 12:

	המשפט לביצוע	num	sum	num%6==0	**פלט**
1	Console.Write("Enter a number: ");	?	?		Enter a number:
2	num = **int**.Parse(Console. ReadLine());	12	?		
3	**if** (num % 6 == 0)	12	?	**true**	
3.1	sum = num * 6;	12	72		
3.2	Console.WriteLine("**…")	12	72		**********************
3.3	Console.WriteLine("You Won {0} shekels", sum);	12	72		* You won 72 shekels *
3.4	Console.WriteLine("**…")	12	72		**********************

סוף פתרון בעיה 3

שאלה 5.14

בנו טבלת מעקב אחר ביצוע התוכנית SheshEsh מפתרון בעיה 3 עבור הקלט 10.

שאלה 5.15

במשתנים x ו-y שמורים שני ערכים מטיפוס שלם.

מטרת התוכנית היא להציג כפלט את המספר הגדול ראשון ואחריו את המספר הקטן.

השלימו את משפט ה-**if** הבא באמצעות ביטוי בוליאני מתאים.

```
if (_________________)
{
```

```
    temp = x;
    x = y;
    y = temp;
}
Console.WriteLine(x, y);
```

שאלה 5.16

פתחו אלגוריתם שהקלט שלו הוא שני מספרים חיוביים. המספר הראשון מציין את משקלו של החתול גארפילד בק״ג והמספר השני מציין את משקלו של הכלב סנופי. אם משקלו של גארפילד גדול ממשקלו של סנופי, האלגוריתם צריך לחשב כמה ק״ג שוקל גארפילד יותר מסנופי ולהציג הודעה מתאימה הכוללת את הערך שחושב. אחרת האלגוריתם צריך לחשב כמה ק״ג שוקל סנופי יותר מגארפילד ולהציג הודעה מתאימה הכוללת את הערך שחושב.

למשל פלט מתאים עבור הקלט 15 10 הוא:

```
Snoopy is heavier than Garfield, the difference is 5 kg
```

פלט מתאים עבור הקלט 13 17 הוא:

```
Garfield is heavier than Snoopy, the difference is 4 kg
```

ישמו את האלגוריתם בשפת C#.

שאלה 5.17

פתחו אלגוריתם שהקלט שלו הוא שני מספרים שלמים. אם המספר הראשון גדול מהשני, האלגוריתם מחשב את סכומם ומציג אותו כפלט. אחרת הוא מחשב את מכפלת המספרים ומציג אותה כפלט. ישמו את האלגוריתם בשפת C#.

ביטויים בוליאניים הכוללים תווים

הביטויים הבוליאניים שראינו עד כה כללו פעולות השוואה על מספרים. עם זאת המספרים השלמים או המספרים הממשיים אינם הטיפוסים היחידים שערכיהם ניתנים להשוואה. בגלל ההתאמה של תווים למספרים שלמים, גם ערכיו של הטיפוס התווי ניתנים להשוואה, כפי שמדגימה הבעיה הבאה.

בעיה 4

מטרת הבעיה ופתרונה: הצגת פעולות השוואה על טיפוס תווי.

בסדרות של סימנים, הכוללות מספר סופי של סימנים, לכל סימן יש סימן עוקב מלבד לאחרון. עבור סדרות כאלה נהוג להגדיר את הסימן הראשון כ״עוקב מעגלית״ לסימן האחרון.
פתחו וישמו אלגוריתם אשר הקלט שלו הוא אות מן האותיות הגדולות של הא״ב האנגלי (A..Z), והפלט שלו הוא האות העוקבת ״בצורה מעגלית״ והודעה מתאימה. עבור אות קלט השונה מהאות Z יהיה הפלט האות הבאה בא״ב והודעה המכריזה שזו האות העוקבת. עבור האות Z הפלט יהיה האות A והודעה המכריזה על חזרה להתחלה.

פירוק הבעיה לתת-משימות

1. קליטת אות
2. חישוב האות העוקבת "בצורה מעגלית"

3. הצגת האות העוקבת בצירוף הודעה מתאימה

בחירת משתנים

נבחר את המשתנים הבאים מטיפוס תווי:

letter – ישמור את האות הניתנת כקלט

nextLetter – ישמור את האות העוקבת "בצורה מעגלית" לאות הקלט

האלגוריתם

1. *קלוט אות ב-*letter
2. *אם ערכו של* letter *הוא* 'Z'
   - *2.1. השם את האות* 'A' *ב-*nextLetter
   - *2.2. הצג כפלט את ההודעה* "Back to start:" *ואת ערכו של* nextLetter
3. *אחרת*
   - *3.1. חשב את האות העוקבת לאות הנתונה ב-*letter *והשם ב-*nextLetter
   - *3.2. הצג כפלט את ההודעה* "The next letter is:" *ואת ערכו של* nextLetter

יישום האלגוריתם

כיצד נבצע את ההשוואה בשורה 2 של האלגוריתם? ניתן להשוות בין ערכים מטיפוס תווי, בדיוק כשם שניתן להשוות בין ערכים מטיפוס שלם או ממשי, בשימוש באופרטור ההשוואה הרגיל של השפה. לכן הביטוי הבוליאני המתאים הוא `letter == 'Z'`.

התוכנית המלאה

```
/*
קלט: אות אנגלית גדולה
פלט: הודעה הכוללת את האות העוקבת "בצורה-מעגלית" לאות הנתונה
*/
using System;
public class NextLetterInCircle
{
    public static void Main ()
    {
        // הצהרה על משתנים בתוכנית
        char letter; // אות הקלט
        char nextLetter; // האות העוקבת
        // קליטת המשתנים
1.      Console.Write("Enter a letter from the ABC: ");
2.      letter = char.Parse(Console.ReadLine());
        // ההוראה לביצוע-בתנאי: חישוב האות העוקבת
3.      if (letter ==  'Z')
        {   // המקרה המיוחד - האות האחרונה
3.1.        nextLetter = 'A';
3.2.        Console.WriteLine("Back to start: {0}", nextLetter);
        } // if
4.      else
        {     // האות איננה האחרונה
4.1.        nextLetter = (char) (letter + 1); //המרת טיפוס (casting)
4.2.        Console.WriteLine("The next letter is: {0}", nextLetter);
        } // else
```

```
    } // Main
} // class NextLetterInCircle
```

מעקב

נעקוב אחר מהלך ביצוע התוכנית NextLetterInCircle עבור הקלט 'C':

	המשפט לביצוע	letter	nextLetter	letter=='Z'	פלט
1	Console.Write("Enter a letter … ");	?	?		Enter a letter …
2	letter = **char**.Parse(Console.ReadLine());	'C'	?		
3	**if** (letter == 'Z')	'C'	?	**false**	
4.1	nextLetter = (**char**) (letter + 1);	'C'	'D'		
4.2	Console.WriteLine("The next letter is: {0}", nextLetter);	'C'	'D'		The next letter is: D

סוף פתרון בעיה 4

שאלה 5.18

בנו טבלת מעקב אחר ביצוע התוכנית NextLetterInCircle לפתרון בעיה 4 עבור הקלט 'Z'.

כזכור, קבוצת התווים של המחשב כוללת גם את הספרות '9',...,'1','0'. כלומר ניתן להתייחס אל ספרה כאל ערך מטיפוס תווי. ספרות עוקבות מסודרות כתווים עוקבים בקבוצת התווים. השאלה הבאה מתייחסת לספרות כאל תווים.

שאלה 5.19

נתון קטע התוכנית הבא:

```
char digit;
char x;
Console.Write("Enter a digit between 0 and 9: ");
digit = char.Parse(Console.ReadLine());
if (digit == '0')
    x = '9';
else
    x = (char)(digit - 1);
Console.WriteLine(x);
```

הקלט לקטע התוכנית הוא ספרה בין '0' ל-'9'.

א. מהו הפלט עבור הקלט '5', ומהו הפלט עבור הקלט '0'?

ב. מהו הקלט אשר הפלט עבורו יהיה 8?

ג. מהי מטרת קטע התוכנית?

מאחר שערכי הטיפוס התווי ניתנים להשוואה, ניתן להשוות ערכים מטיפוס תווי גם באמצעות הסימנים >, <, =>, =<, ולא רק באמצעות הסימנים == ו-=!.

למשל נניח שערכי המשתנים מטיפוס תווי let1 ו-let2 הם 'E' ו-'T' בהתאמה. אז ערכו של כל אחד מן הביטויים הבוליאניים הבאים הוא **true**: 'A' < 'B', '7' > '3', let1 >= 'E' ו-let1 <= let2.

ערכיו של הטיפוס התווי ניתנים להשוואה. לכן ניתן להשתמש בכל **פעולות ההשוואה על תווים** בדומה לשימושן על ערכים מספריים.

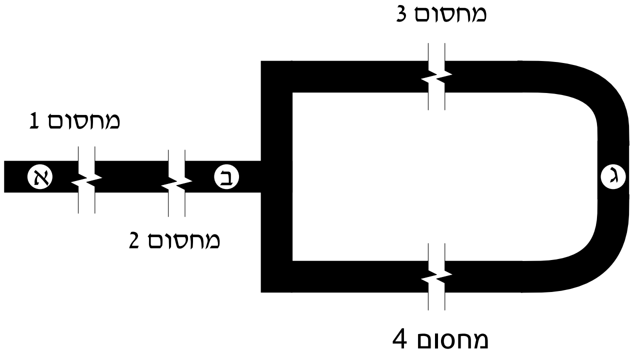
שאלה 5.20

פתחו אלגוריתם אשר הקלט שלו הוא שתי אותיות שונות מן הא״ב האנגלי, והוא מציג את אותיות הקלט פעמיים: בשורה אחת ב״סדר עולה״ ובשורה הבאה ב״סדר יורד״. ״סדר עולה״ פירושו: האות שמופיעה קודם בא״ב האנגלי תוצג משמאל, והאות האחרת תוצג מימינה. ״סדר יורד״ הוא סדר הפוך ל״סדר עולה״. ישמו את האלגוריתם בשפת C#.

להעמקה בתבנית ***סידור ערכים בסדרה*** פנו לסעיף התבניות המופיע בסוף הפרק.

5.2 תנאי מורכב

בסעיף זה נכיר תנאים מורכבים. תנאים מורכבים הם תנאים הבנויים מקישור של תנאים פשוטים יותר. נבחן את השרטוט הבא המתאר מערכת כבישים:



איור 5.1 – מערכת כבישים

במערכת הכבישים המתוארת באיור 5.1 ניתן להגיע מנקודה א לנקודה ג. כדי לעשות זאת יש להגיע מנקודה א לנקודה ב, ומנקודה ב לנקודה ג. על הכבישים נמצאים מחסומים. כדי להגיע מנקודה א לנקודה ב יש לעבור במחסום 1 ובמחסום 2. כלומר, רק אם אפשר לעבור בשני המחסומים, אפשר להגיע מנקודה א לנקודה ב. ניתן לתאר זאת כך:

***אם** מחסום 1 מורם **וגם** מחסום 2 מורם*
  ניתן להגיע מנקודה א לנקודה ב
אחרת
  לא ניתן להגיע מנקודה א לנקודה ב

תנאי המעבר מנקודה א לנקודה ב מתואר ב**תנאי מורכב**, שהוא קישור התנאי *מחסום 1 מורם* אל התנאי *מחסום 2 מורם* באמצעות המילה ***וגם***.

נתבונן כעת באפשרות להגיע מנקודה ב לנקודה ג. כדי לעשות זאת יש לעבור במחסום 3 או במחסום 4. כלומר, אם אפשר לעבור באחד המחסומים (או בשניהם) אפשר להגיע מנקודה ב לנקודה ג. ניתן לתאר זאת כך:

***אם** מחסום 3 מורם **או** מחסום 4 מורם*
  ניתן להגיע מנקודה ב לנקודה ג
אחרת
  לא ניתן להגיע מנקודה ב לנקודה ג

תנאי המעבר מנקודה ב לנקודה ג מתואר על ידי תנאי מורכב, שהוא קישור התנאי *מחסום 3 מורם* אל התנאי *מחסום 4 מורם* באמצעות המילה *וגם*.

וגם ו-*או* הם קְשָרִים לוגיים, המאפשרים ליצור מביטויים בוליאניים פשוטים ביטויים בוליאניים מורכבים.

הקֶשֶר *וגם*

ראשית נתמקד בקשר הראשון מבין השניים שהודגמו בניתוח של איור 5.1.

בעיה 5

מטרת הבעיה ופתרונה: הצגת תנאי מורכב הכולל את הקשר *וגם*.

סדרת מספרים נקראת "סדרה עולה ממש" אם ערכו של כל איבר בסדרה קטן ממש מערכו של האיבר הבא אחריו. כלומר ערכו של האיבר הראשון בסדרה קטן ממש מערכו של האיבר השני, ערכו של האיבר השני קטן ממש מערכו של האיבר השלישי, וכן הלאה. למשל סדרת המספרים 10 7 2 1 היא סדרה עולה ממש, וסדרת המספרים 7 3 1 1 איננה סדרה עולה ממש. פתחו אלגוריתם אשר הקלט שלו הוא סדרה של שלושה מספרים שלמים, והפלט שלו הוא הודעה האם סדרת המספרים היא סדרה עולה ממש. אם הסדרה עולה ממש, יש לצרף להודעה את סדרת ההפרשים שבין איברי הסדרה המקורית. כלומר את ההפרש בין המספר השני לראשון, ואת ההפרש בין המספר השלישי לשני. ישמו את האלגוריתם בשפת התכנות C#.

פירוק הבעיה לתת-משימות

1. קליטה של שלושת איברי הסדרה.
2. בדיקה אם הסדרה עולה ממש והצגת הודעה מתאימה.

בחירת משתנים

נבחר שלושה משתנים מטיפוס שלם לשמירת שלושת איברי הסדרה:

num1, **num2**, **num3** – ישמרו את שלושת מספרי הסדרה הנתונה.

האלגוריתם

? בכתיבת האלגוריתם עלינו לנסח תנאי אשר יתקיים כאשר הסדרה הנתונה עולה ממש, ולא יתקיים כאשר הסדרה איננה עולה ממש. מה יהיה התנאי המתאים?

התנאי אשר יתקיים כשהסדרה עולה ממש יהיה:

המספר השלישי גדול מהמספר השני וגם המספר השני גדול מהמספר הראשון

ניתן לכתוב זאת גם כך:

num3>num2 *וגם* num2>num1

שימו ♥: אופן הכתיבה הבא אינו חוקי: num3 > num2 > num1

תנאי זה הוא תנאי מורכב, הכולל קישור באמצעות המילה *וגם* בין שני התנאים.

נציג אלגוריתם לפתרון הבעיה, תוך שימוש בתנאי שניסחנו:

1. קלוט שלושה איברי סדרה ב-num1, ב-num2 וב-num3
2. אם num3>num2 וגם num2>num1
   2.1. הצג כפלט הודעה כי הסדרה עולה ממש
   2.2. הצג כפלט את ערך ההפרש num2-num1 ואת ערך ההפרש num3-num2
3. אחרת
   3.1. הצג כפלט הודעה כי הסדרה אינה עולה ממש

שימו ♥: באלגוריתם מופיעה הוראת פלט שכוללת ערכי הפרשים בין משתנים. לא בחרנו משתנים לשמירת ההפרשים, אלא כללנו בהוראת הפלט ביטויים המבטאים את ההפרשים.

יישום האלגוריתם

ב-#C, הקשר *וגם* נכתב באמצעות הסימן `&&`.

```
/*
קלט: 3 מספרים שלמים
פלט: הודעה אם סדרת המספרים היא סדרה עולה ממש
*/
using System;
public class CheckSequence
{
    public static void Main ()
    {
        // הצהרה על משתנים בתוכנית
        int num1, num2,num3;
        // קליטת המשתנים
1.      Console.Write("Enter first number: ");
2.      num1 = int.Parse(Console.ReadLine());
3.      Console.Write("Enter second number: ");
4.      num2 = int.Parse(Console.ReadLine());
5.      Console.Write("Enter third number: ");
6.      num3 = int.Parse(Console.ReadLine());
7.      if ((num3 > num2) && (num2 > num1))
        {
            // הסדרה עולה ממש
7.1.        Console.WriteLine("The sequence of numbers is " +
                                                " strongly increasing");
7.2.        Console.WriteLine("The sequence of differences " +
                          "is {0} {1}", (num2 - num1), (num3 - num2));
        }
8.      else
8.1.        Console.WriteLine("The sequence of numbers is not " +
                                                " strongly increasing");
    } // Main
} // class CheckSequence
```

מעקב

נעקוב אחר מהלך ביצוע התוכנית עבור הקלט 4 2 1:

מספר שורה	המשפט לביצוע	num1	num2	num3	Num3>num2	num2>num1	פלט
1	`Console.Write("Enter first number: ");`	?	?	?			Enter first number:
2	`num1 = int.Parse( Console.ReadLine());`	1	?	?			
3	`Console.Write("Enter second number: ");`	1	?	?			Enter second number:
4	`num2 = int.Parse( Console.ReadLine());`	1	2	?			
5	`Console.Write("Enter third number: ");`	1	2	?			Enter third number:
6	`num3 = int.Parse( Console.ReadLine());`	1	2	4			
7	`if ((num3 > num2) && (num2 > num1))`	1	2	4	**true**	**true**	
7.1	`Console.WriteLine( "The sequence of numbers is strongly increasing");`	1	2	4			The sequence of numbers is strongly increasing
7.2	`Console.WriteLine("The sequence of differences is ... ");`	1	2	4			The sequence of differences is 1 2

סוף פתרון בעיה 5

תנאי מורכב הוא תנאי המורכב מקישור בין תנאים פשוטים (ביטויים בולאניים פשוטים). עד כה ראינו דרך אחת ליצור תנאי מורכב על ידי קישור תנאים פשוטים: באמצעות הקשר ***וגם*** (and), אשר משמעותו היא שהתנאי המורכב מתקיים רק כאשר שני התנאים הפשוטים מתקיימים ביחד בו-זמנית.

הטבלה הבאה מתארת את ערכיו של ביטוי בוליאני המורכב מקשר ***וגם*** בין שני ביטויים בולאניים. הטבלה נקראת **טבלת אמת של קשר *וגם***:

ביטוי 2 \ ביטוי 1	**false**	**true**
false	**false**	**false**
true	**false**	**true**

כפי שניתן לראות בטבלה, ערכו של הביטוי הבוליאני המורכב הוא **true** (אמיתי, נכון) **רק** כאשר ערך שני הביטויים 1 ו-2, הוא **true**. בכל מקרה אחר, ערכו של הביטוי הבוליאני המורכב הוא **false** (שקרי, לא נכון).

הקשר **וגם** מיושם ב-C# בסימן הפעולה `&&`. יש להקיף בסוגריים את הביטויים הבוליאניים המקושרים ב-`&&`.

בטבלת מעקב אחר מהלך ביצוע תוכנית שבה משפט **if** הכולל ביטוי בוליאני מורכב, נקצה עמודה לכל אחד מן הביטויים הבוליאניים הפשוטים המרכיבים את הביטוי הבוליאני המורכב.

שאלה 5.21

בתוכנית `CheckSequence` לפתרון בעיה 5 מופיע הביטוי הבוליאני המורכב הבא:

```
((num3 > num2) && (num2 > num1))
```

השלימו את הטבלה הבאה:

num1	num2	num3	num3 > num2	num2 > num1	(num3 > num2)&& (num2 > num1)	**פלט**
1	2	2				
1	1	2				
1	1	1				
0	1	2				

שאלה 5.22

השלימו את התנאי המורכב המתאים בכל אחת מן ההוראות הבאות:

א. אם האות הראשונה שווה לרביעית וגם ______________________
המילה בת ארבע האותיות היא פלינדרום

ב. אם שנת הלידה שלך גדולה מ-1985 וגם ______________________
שנת הלידה שלך היא בין 1985 ל-1995

ג. אם let>='A' וגם ______________________
let מייצג אות אנגלית גדולה

שאלה 5.23

כתבו את התנאים הבאים המנוסחים במילים, כביטויים בוליאניים:

א. ערכו של המשתנה `x` גדול מ-0 וקטן מ-50.

ב. ערכו של המשתנה `let` אינו התו 'a' ואינו התו 'z'.

ג. הערך המוחלט של הפרשי ערכי המשתנים `x` ו-`y` גדול מערכו של `x` וקטן מערכו של `y`.

שאלה 5.24

ביטוי בוליאני מורכב יכול לכלול יותר משני ביטויים בוליאניים פשוטים.

במשתנים `temp1`, `temp2` ו-`temp3` ערכים כלשהם.

א. כתבו ביטוי בוליאני המבטא כי ערכי המשתנים `temp1`, `temp2` ו-`temp3` שונים זה מזה. האם נחוצים יותר משני ביטויים בוליאניים פשוטים לכתיבת הביטוי?

ב. כתבו ביטוי בוליאני המבטא כי ערכי המשתנים `temp1`, `temp2` ו-`temp3` שווים זה לזה. האם נחוצים יותר משני ביטויים בוליאניים פשוטים לכתיבת הביטוי?

שאלה 5.25

עבור כל אחד מן הביטויים הבוליאניים המורכבים הבאים, תנו דוגמה לערך של המשתנה `num` או `let` אשר עבורו יהיה ערכו של הביטוי הבוליאני **true**, ותנו דוגמה לערך אשר עבורו יהיה ערכו של הביטוי הבוליאני **false**.

ביטוי	true	false
`(num >= 0) && (num <= 5)`		
`(num > 0) && (num != 1)`		
`(num > 2) && ((num % 2) == 0)`		
`(let != 'z') && (let > 'x')`		

שאלה 5.26

פתחו אלגוריתם אשר הקלט שלו הוא שלושה תווים, והפלט שלו הוא התו העוקב לגדול מבין התווים, אם שלושת התווים הם תווים עוקבים ונתונים בסדר עולה.

למשל עבור הקלט B C D יהיה הפלט E, ועבור הקלט A C D יהיה הפלט ריק (כלומר לא יוצג דבר, כיוון שהתווים אינם תווים עוקבים).

ישמו את האלגוריתם בשפת התכנות C#.

כתבו את חישוב התו העוקב כביטוי במשפט הפלט.

להעמקה בתבנית ***ערכים עוקבים?*** פנו לסעיף התבניות המופיע בסוף הפרק.

הקֶשֶר *או*

בניתוח של איור 5.1 הזכרנו קשר נוסף – הקשר *או*. נדון כעת בשימוש בקשר זה בכתיבת אלגוריתמים וביישומם.

בעיה 6

מטרת הבעיה ופתרונה: הצגת תנאי מורכב הכולל את הקשר *או*.

פתחו אלגוריתם שהקלט שלו הוא שני מספרים שלמים חיוביים דו ספרתיים, והפלט שלו הוא הודעה אם שני המספרים מורכבים מאותן ספרות.

למשל, עבור הקלט: 19 91, הפלט יהיה הודעה שהמספרים מורכבים מאותן ספרות, ועבור הקלט 19 81, הפלט יהיה הודעה שהמספרים אינם מורכבים מאותן ספרות.

ישמו את האלגוריתם בשפת התכנות C#.

ניתוח הבעיה בעזרת דוגמאות

נבחן מספר דוגמאות קלט מגוונות אשר כוללות את המספר 91:

עבור כל אחד מן הקלטים: 91 91, 91 19, 19 91 תוצג כפלט הודעה שהמספרים מורכבים מאותן ספרות.

עבור כל קלט אחר הכולל את המספר 91, מלבד שלושת הקלטים האלה, תוצג כפלט הודעה שהמספרים אינם מורכבים מאותן ספרות.

מהתבוננות בשלושת הקלטים המתוארים ניתן לראות שתוצג כפלט הודעה שהמספרים מורכבים מאותן ספרות אם ורק אם המספר הנוסף ל-91 הוא 91 עצמו או שהוא 19, כלומר, המספר המתקבל מ-91 על ידי היפוך סדר הספרות.

פירוק הבעיה לתת-משימות

נוכל לנסח רעיון ראשוני לפתרון: ראשית נפרק את המספר הראשון לספרותיו ונבנה את המספר המתקבל ממנו לאחר היפוך בסדר הספרות. כעת ניתן לבדוק אם המספר השני שווה למספר הראשון או למספר ההפוך לראשון. אם המספר השני שווה לאחד משניהם אז תוצג כפלט הודעה שהמספרים מורכבים מאותן ספרות.

נפרק לתת-משימות על פי הרעיון שהצענו:

1. קליטת שני מספרים שלמים חיוביים דו-ספרתיים
2. פירוק המספר הראשון לספרותיו
3. הרכבת מספר חדש שמתקבל מהמספר הראשון לאחר היפוך סדר הספרות
4. השוואת המספר השני למספר הראשון ולמספר החדש
5. הצגת הודעת פלט מתאימה

בחירת משתנים

נבחר משתנים מטיפוס שלם על פי התת-משימות המתוארות:

num1 – לשמירת המספר הראשון
num2 – לשמירת המספר השני
tens – לשמירת ספרת העשרות של num1
units – לשמירת ספרת האחדות של num1
invNum1 – לשמירת המספר ההפוך בסדר ספרותיו ל-num1

האלגוריתם

❓ כיצד נבנה את המספר ההפוך למספר הראשון בסדר ספרותיו?

זוהי תבנית ***בניית מספר*** דו-ספרתי, המוכרת לנו מפרק 4: ספרת העשרות של המספר החדש היא ספרת האחדות של המספר הראשון. ספרת האחדות של המספר החדש היא ספרת העשרות של המספר הראשון. לכן יש להכפיל את ערכו של units ב-10 ולחבר למכפלה את ערכו של tens.

❓ בכתיבת האלגוריתם עלינו לנסח תנאי אשר יתקיים כאשר num2 יהיה שווה לאחד המספרים הנתונים ב-num1 וב-invNum1. מהו התנאי המתאים?

התנאי הוא: num2 == num1 ***או*** num2 == invNum1

תנאי זה הוא תנאי מורכב הכולל קישור בין שני תנאים באמצעות המילה ***או***.

❓ האם ייתכן שיתקיימו בו-זמנית שני התנאים הפשוטים הכלולים בתנאי המורכב שניסחנו?

כן, זה ייתכן. למשל אם הקלט הוא 11 11. במקרה כזה המספר השני שווה גם למספר הראשון, וגם למספר שמתקבל מהמספר הראשון בהיפוך סדר הספרות. גם במקרה כזה, אנחנו מעוניינים כמובן שהתנאי המורכב יתקיים. כלומר התנאי המורכב מתקיים כאשר num2 שווה לאחד המספרים הנתונים ב-num1 וב-invNum1, או לשניהם.

יישום האלגוריתם

ב-C#, הקשר **או** נכתב באמצעות הסימן || (שני קווים אנכיים רצופים).

```
/*
קלט: 2 מספרים חיוביים שלמים
 פלט: הודעה אם המספרים מורכבים מאותן ספרות
*/
using System;
public class DigitEquality {
      public static void Main()
      {
        // הצהרה על משתנים בתוכנית
        int num1;       // מספר ראשון
        int num2;       // מספר שני
        int tens;       // ספרת העשרות של המספר הראשון
        int units;      // ספרת האחדות של המספר הראשון
        int invNum1;    // המספר השני הפוך
        // קליטת המשתנים
1.      Console.Write("Enter first number: ");
2.      num1 = int.Parse(Console.ReadLine());
3.      Console.Write("Enter second number: ");
4.      num2 = int.Parse(Console.ReadLine());
       //חישוב ספרת האחדות והעשרות וחישוב המספר ההפוך
5.      tens = num1 / 10;
6.      units = num1 % 10;
7.      invNum1 = units * 10 + tens;
        //בדיקה אם המספר השני שווה למספר הראשון או למספר ההפוך לו
8.      if ((num2 == num1) || (num2 == invNum1))
8.1.        Console.WriteLine("The numbers have the same digits");
9.      else
9.1.        Console.WriteLine("The numbers don't have the" +
                                                  " same digits");
    }// Main
}// class DigitEquality
```

סוף פתרון בעיה 6

דרך נוספת ליצור תנאי מורכב היא לקשר תנאים פשוטים באמצעות הקשר **או** (or). משמעותו היא שהתנאי המורכב מתקיים כאשר לפחות אחד משני התנאים הפשוטים מתקיים.

הטבלה הבאה מתארת את ערכיו של ביטוי בוליאני המורכב מקשר **או** בין שני ביטויים בוליאניים. הטבלה נקראת **טבלת אמת של קשר או**:

ביטוי 1 / ביטוי 2	**false**	**true**
false	**false**	**true**
true	**true**	**true**

כפי שניתן לראות בטבלה, ערכו של הביטוי הבוליאני המורכב הוא **true** כאשר לפחות ערך אחד משני הביטויים 1 או 2, הוא **true**. רק אם ערכי שני הביטויים הם **false** אז ערכו של הביטוי הבוליאני המורכב הוא **false**.

הקשר *או* מיושם ב-#C בסימן הפעולה `||`. יש להקיף בסוגריים את הביטויים הבוליאניים המקושרים ב-`||`.

שאלה 5.27

בתוכנית `DigitEquality` לפתרון בעיה 6 מופיע הביטוי הבוליאני המורכב הבא:

```
(num1 == num2) || (num2 == invNum1)
```

ציינו עבור כל אחד מן הקלטים הבאים: מהו ערכו של הביטוי הפשוט השמאלי, מהו ערכו של הביטוי הפשוט הימני, מהו ערכו של הביטוי המורכב, ומהו הפלט במהלך ביצוע התוכנית.

קלט	num1 == num2	num2 == invNum1	(num1 == num2) || (invNum1 == num2)	**פלט**
25 56				
25 25				
25 52				
55 55				

שאלה 5.28

השלימו את התנאי המורכב המתאים בכל אחת מן ההוראות הבאות:

א. *אם האות הראשונה היא* A *או* __________________
לפחות אחת האותיות היא A *במילה בת שתי אותיות*

ב. *אם גובה נמוך מ-18 או* ______________________
גובה איננו הגובה המקובל של חייל בשירות סדיר

ג. *אם* num > 10 *או* _________________________
num *שומר ערך שהוא גדול מ-10 או קטן מ-5*

שאלה 5.29

כתבו את התנאים הבאים המנוסחים במילים כביטויים בוליאניים.

א. ערכו של המשתנה `x` חיובי או ערכו של המשתנה `y` הוא התו 'A'.
ב. ערכו של המשתנה `x` קטן מ-1 או גדול מ-7.
ג. ערכו של המשתנה `x` זוגי או מתחלק ב-3 ללא שארית.

להעמקה בתבנית ***זוגיות מספר*** פנו לסעיף התבניות המופיע בסוף הפרק.

להעמקה בתבנית ***מחלק של?*** פנו לסעיף התבניות המופיע בסוף הפרק.

שאלה 5.30

במשתנים `side1`, `side2` ו-`side3` שמורים אורכי שלוש צלעות של משולש.

א. כתבו ביטוי בוליאני המבטא שהמשולש שווה שוקיים (במשולש שווה שוקיים לפחות שתי צלעות שוות).

ב. כתבו ביטוי בוליאני המבטא שהמשולש ישר זווית (במשולש ישר זווית סכום ריבועי שני הניצבים שווה לריבוע היתר – משפט פיתגורס).

שאלה 5.31

עבור כל אחד מן הביטויים הבוליאניים המורכבים הבאים, תנו דוגמה לערך של המשתנה num אשר עבורו יהיה ערכו של הביטוי הבוליאני **true**, ותנו דוגמה לערך אשר עבורו יהיה ערכו של הביטוי הבוליאני **false**.

ביטוי	ערכו של הביטוי false	ערכו של הביטוי true
`num != 0`	num =	num =
`(num < 2)||(num > 2)`	num =	num =
`(num > 0 )||(num == -5)`	num =	num =
`((num%2) == 0)||(num < 0)`	num =	num =

שאלה 5.32

לפעמים ניתן לצמצם ביטוי בוליאני מורכב לביטוי פשוט. בהנחה ש-num מייצג מספר שלם צמצמו כל אחד מן הביטויים המורכבים לביטוי פשוט (כלומר ללא שימוש בקשרים):

א. `(num < -1) || (num > 1)` ________________________

ב. `(num > -1) && (num < 1)` ________________________

שאלה 5.33 (מבגרות 2003)

לפניכם קטע תוכנית:

```
a = int.Parse(Console.ReadLine());
b = int.Parse(Console.ReadLine());
if ( a < b || a < 100 )
    Console.WriteLine("The expression value: true");
else
    Console.WriteLine("The expression value: false");
```

בחרו במספר שייקלט ב-a ובמספר שייקלט ב-b, כך שיתקבל הפלט

```
The expression value: false
```

נמקו את בחירתכם.

שאלה 5.34

פתחו אלגוריתם אשר הקלט שלו הוא תו. האלגוריתם בודק אם התו הוא תו חוקי לניחוש בטופס ספורטוטו (כלומר 1, 2 או x). האלגוריתם מציג הודעה מתאימה כפלט. ישמו את האלגוריתם בשפת C#.

שאלה 5.35

בתחרות קליעת כדור לארגז, קולעים כדור לארגז שאורכו מטר אחד. תחילתו של הארגז היא במרחק 10 מטרים מן הקולע.

יש לפתח אלגוריתם אשר הקלט שלו הוא מרחק נפילת הכדור מהקולע והפלט שלו הוא הודעה אם הכדור נכנס לארגז. אם הכדור לא נכנס לארגז, יש לצרף לפלט גם את המרחק בין מקום נפילת הכדור ל**מרכז** הארגז.

למשל, עבור הקלט 10.3 הפלט הוא: נכנס.

ועבור הקלט 12 הפלט הוא: לא נכנס, החטאת את מרכז הארגז ב-1.5 מ'.

נתחו תחילה את הבעיה (בעזרת דוגמאות קלט מייצגות) ובחרו משתנים. לאחר מכן:

א. נסחו תנאי מורכב באמצעות קשר ***וגם*** שיתקיים כאשר הכדור נכנס לארגז.

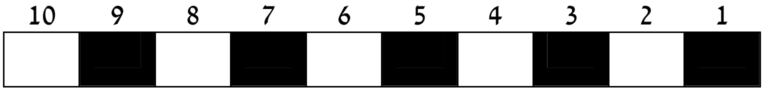
ב. נסחו תנאי מורכב באמצעות קשר ***או*** שיתקיים כאשר הכדור **לא** נכנס לארגז.

ג. נסחו את התנאי שבסעיף ב כתנאי לא מורכב.

ד. כתבו את האלגוריתם לפתרון הבעיה (בחרו את אחד מהתנאים שבסעיפים א-ג), וישמו את האלגוריתם בתוכנית בשפת C#.

שאלה 5.36

נתון הלוח הבא ובו עשר משבצות הממוספרות מ-1 עד 10:



במשתנה x שמור מספר שלם בין 1 ל-10 המבטא את מקום הכלי X על הלוח. במשתנה y שמור מספר בין 1 ל-10 המבטא את מקום הכלי Y על הלוח.

א. כתבו ביטוי בוליאני שערכו **true** אם הכלי X נמצא בחצי השמאלי של הלוח (כלומר על אחת מחמש המשבצות השמאליות).

ב. השלימו את תיאור המשמעות של קיום התנאי ואת הפלט המתאים במשפט ה-**if** הבא:

```
if (Math.Abs(x - y) == 1) // ______________________________
    Console.WriteLine("_____________________________________");
else // ____________________________________________________
    Console.WriteLine("_____________________________________");
```

ג. הביטוי הבוליאני הבא אמור לבטא מצבים שבהם הכלי X נמצא מימין לכלי Y על הלוח:

```
x != y
```

הביטוי שגוי. תקנו אותו.

ד. כתבו ביטוי בוליאני פשוט (לא מורכב) שערכו **true** אם הכלי Y נמצא על משבצת שחורה. שימו לב שבביטוי עליכם לבטא את המשותף לחמש המשבצות השחורות.

ה. השלימו את תיאורי המשמעות של קיום התנאי ושל אי-קיומו ואת הפלט המתאים במשפט ה-**if** הבא:

```
if (((x-y) % 2)  == 0) // __________________________
    Console.WriteLine("______________________________");
else // _____________________________________________
    Console.WriteLine("______________________________");
```

ו. הביטוי הבוליאני הבא אמור לבטא מצבים שבהם שני הכלים נמצאים על משבצות בצבעים שונים:

```
((x * y) % 2 == 0)
```

הביטוי נכון רק עבור חלק מן המקרים האפשריים. עבור אילו מקרים הביטוי נכון, ועבור אילו מקרים איננו נכון? כתבו ביטוי שיהיה נכון עבור כל המקרים האפשריים.

תנאים מורכבים מעורבים

בעזרת הקשרים *וגם* ו-*או* ניתן ליצור ביטויים מורכבים אף יותר מאלו שראינו עד כה, אשר מערבים את שני הקשרים.

שאלה 5.37

עבור כל אחד מן הביטויים הבוליאניים המורכבים הבאים, תנו דוגמה לערך של המשתנה num אשר עבורו יהיה ערכו של הביטוי הבוליאני **true**, ותנו דוגמה לערך אשר עבורו יהיה ערכו של הביטוי הבוליאני **false**.

ביטוי	ערכו של הביטוי false	ערכו של הביטוי true
`((num != 0) && (num >= 8)) || (num == 3)`	num =	num =
`((num<2)||(num>2))&&((num<7)||(num!=1))`	num =	num =
`(num == 0) && ((num > 0 ) ||(num == -5))`	num =	num =

בדומה לקדימות המוגדרת ביחס לפעולות חשבוניות (כגון פעולת כפל קודמת לפעולת חיבור), מוגדרת גם **קדימות** ביחס לקשרים בוליאניים: הקשר `&&` קודם לקשר `||`.

בכל זאת, לשם בהירות התוכנית ולשם קריאותה עדיף להשתמש בסוגריים כדי להבהיר את טווח הפעולה של כל קשר.

שאלה 5.38 (מבגרות 2003)

נתון הביטוי הבוליאני: `(z > x) || (x < 0) && (z - y > 9)`

מהו הערך של הביטוי עבור הנתונים: x= -2, y= 5, z= 13? פרטו את כל שלבי החישוב.

5.3 קינון של הוראה לביצוע-בתנאי

בסעיף זה נראה כי לעתים נוח ליצור הוראה מורכבת לביצוע-בתנאי. זוהי הוראה לביצוע-בתנאי שאחת (או יותר) מהוראותיה היא עצמה הוראה לביצוע-בתנאי.

בעיה 7

מטרת הבעיה ופתרונה: הצגת הוראה מקוננת לביצוע-בתנאי.

באולימפיאדת החיות מתקיימת תחרות ריצה למרחק 100 מטר. צבי אשר רץ 100 מטר בזמן של 10 **שניות** או פחות נחשב לצבי מהיר. צב שרץ 100 מטר בזמן של 10 **דקות** או פחות נחשב לצב מהיר.

פתחו אלגוריתם אשר הקלט שלו הוא סוג החיה, T (Turtle) עבור צב ו-D (Deer) עבור צבי, ומספר המציין זמן ריצה בשניות. האלגוריתם יציג הודעה אם החיה שנקלטה מהירה או לא. ישמו את האלגוריתם בשפת C#.

פירוק הבעיה לתת-משימות

1. קליטת שם החיה
2. קליטת תוצאת הריצה בשניות
3. הצגת הודעה אם החיה מהירה או לא

בחירת משתנים

animalType – משתנה מטיפוס תווי שישמור את שם החיה
scoreInSeconds – משתנה מטיפוס שלם שישמור את תוצאת הריצה בשניות

האלגוריתם

? כיצד נחליט אם החיה מהירה?

יש לבדוק תחילה אם החיה היא צבי או צב ואז להשוות את תוצאת הריצה שלה לזמן המגדיר צבי מהיר או צב מהיר, בהתאמה.

נכתוב זאת בצורה הבאה:

אם סוג החיה הוא 'D' *// צבי*
　השווה את תוצאת הריצה ל-10 שניות
אחרת // צב
　השווה את תוצאת הריצה ל-600 שניות

תיארנו כאן מבנה של *אם...אחרת...* שהוא מבנה של ביצוע-בתנאי. אבל כדי לדעת אם הצבי מהיר ואם הצב מהיר יש לכלול בכל אחד מחלקי ההוראה לביצוע-בתנאי הוראה נוספת לביצוע-בתנאי. המבנה המתקבל הוא של *אם...אחרת...* בתוך *אם...אחרת...*, כלומר קינון של הוראות לביצוע-בתנאי.

נציג אלגוריתם לפתרון הבעיה, תוך שימוש בתנאי שניסחנו:

*1. קלוט את שם החיה ב-*animalType
*2. קלוט את תוצאת הריצה של החיה ב-*scoreInSeconds
3. אם animalType == 'D' *// צבי*
　3.1. אם scoreInSeconds <= *10*
　　3.1.1. הצג הודעה כי הצבי מהיר
　3.2. אחרת
　　3.2.1. הצג הודעה כי הצבי אינו מהיר
4. אחרת // צב
　4.1. אם scoreInSeconds <= *600*
　　4.1.1. הצג הודעה כי הצב מהיר
　4.2. אחרת
　　4.2.1. הצג הודעה כי הצב אינו מהיר

שימו לב לאופן הזחת השורות (הזזתן פנימה, אינדנטציה) בכתיבה המקוננת של ההוראות לביצוע-בתנאי. אמנם אין השפה מחייבת זאת, אך מומלץ מאוד להקפיד על כך, משום שכך נוח לשייך כל *אחרת* ל-*אם* המתאים. בכך אנו הופכים את התוכנית לקריאה ולברורה יותר.

יישום האלגוריתם

```
/*
קלט: תו המזהה חיה ותוצאת ריצתה למאה מטרים
פלט: הודעה המציינת אם החיה מהירה או לא
*/
using System;
public class AnimalOlympics
{
    public static void Main()
    {
      // הצהרה על משתנים בתוכנית
      char animalType;
      int scoreInSeconds;
      const int DEER_LIMIT = 10;
      const int TURTLE_LIMIT = 600;
      // קליטת המשתנים
1.    Console.Write("Enter the animal type - D for a deer and " +
                                        "T for a turtle: ");
2.    animalType = char.Parse(Console.ReadLine());
3.    Console.Write("Enter the animal score in seconds: ");
4.    scoreInSeconds = int.Parse(Console.ReadLine());
5.    if (animalType == 'D')   // אם צבי
      {
5.1.      if (scoreInSeconds <= DEER_LIMIT)// האם מהיר לפי ההגדרה
                                         // המתאימה לצבי
5.1.1.        Console.WriteLine("The deer is fast");
5.2.      else
5.2.1.        Console.WriteLine("The deer is not fast");
      } // if animalType
6.    else // צב
      {
6.1.      if (scoreInSeconds <= TURTLE_LIMIT) // האם מהיר לפי
                                           // ההגדרה המתאימה לצב
6.1.1.        Console.WriteLine("The turtle is fast");
6.2.      else
6.2.1.        Console.WriteLine("The turtle is not fast");
      } // else animalType
   }//Main
}// class AnimalOlympics
```

שאלה 5.39

בנו טבלת מעקב אחר ביצוע התוכנית `AnimalOlympics` עבור הקלט 700 T ועבור הקלט 5 D.
חשבו לאילו תחומים במבנה המקונן של משפטי ה-**`if`** בתוכנית, ינותב מהלך הביצוע עבור כל סוג קלט אפשרי?

סוף פתרון בעיה 7

לצורך פתרון בעיה 7 היה מתאים להשתמש במבנה מקונן של הוראות לביצוע-בתנאי.
ההוראה לביצוע-בתנאי שתלויה בסוג החיה (ובתוכנית תלויה במשתנה `animalType`) היא הוראה לביצוע-בתנאי **חיצונית** (בתוכנית היא יושמה במשפט **`if`** חיצוני). הוראה זו מכילה שתי

הוראות לביצוע-בתנאי **פנימיות** (בתוכנית – שני משפטי **if** פנימיים): האחת מוכלת בתחום ה-*אם* של ההוראה החיצונית והשנייה מוכלת בתחום ה-*אחרת* של ההוראה החיצונית. ההוראות הפנימיות הן אלו שביצוען תלוי בתוצאת הריצה (ובתוכנית תלוי בערך המשתנה `scoreInSeconds`).

לעתים, כאשר יש לנתב את מהלך הביצוע של אלגוריתם לאחת מבין שלוש או יותר אפשרויות התלויות במספר תנאים, מתאים להשתמש במבנה מקונן של הוראות לביצוע-בתנאי.

מבנה מקונן (nesting) של ביצוע-בתנאי (משפטי **if**) כולל הוראה לביצוע-בתנאי, אשר אחת או יותר מבין הוראותיה הפנימיות היא בעצמה הוראה לביצוע-בתנאי. למשל המבנה הבא הוא מבנה מקונן של ביצוע-בתנאי:

בכתיבה אלגוריתמית:	**ב-#C:**
אם...	`if(. . .)`
	`{`
...	`. . .`
אם...	`if (. . .)`
...	`{. . .}`
אחרת	`else`
...	`{. . .}`
...	`. . .`
	`}`
אחרת	`else`
	`{`
...	`. . .`
אם...	`if (. . .)`
...	`{. . .}`
אחרת	`else`
...	`{. . .}`
...	`. . .`
	`}`

בכתיבת מבנה מקונן נקפיד על הזחות מתאימות. באלגוריתם ייכתב כל *אחרת* מתחת ל-*אם* המתאים לו. ב-#C ייכתב כל **else** מתחת ל-**if** המתאים לו.

שימו ♥: ההגדרה של קינון הוראות לביצוע-בתנאי היא כללית למדי, ולכן ייתכנו מבנים מקוננים בצורות שונות. כל אחת מההוראות הפנימיות של הוראה לביצוע-בתנאי יכולה להיות בעצמה הוראה לביצוע-בתנאי. לכן ייתכן למשל משפט **if** שתחום ה-**if** שלו מכיל שתי הוראות לביצוע-בתנאי.

בנוסף, הוראה פנימית לביצוע-בתנאי יכולה להיות הוראה לביצוע-בתנאי מכל סוג שהוא – במבנה *אם*... *אחרת*..., במבנה *אם*... או אפילו הוראה לביצוע-בתנאי מקוננת בעצמה.

לכן מהרגע שקינון מצטרף למשחק, המבנים המתקבלים יכולים להיות מורכבים מאוד. משום כך, חשוב לזכור ולהקפיד על הערות המתארות משמעות של קיום תנאי ושל אי-קיומו. הערות אלו מסייעות לנו להבין מבנה מקונן של הוראות לביצוע-בתנאי. הערה המתארת משמעות של **קיום** תנאי במשפט **if** מבהירה בשפה ברורה וקריאה את הסיבה לניתוב מהלך הביצוע של משפט ה-**if** אל תחום ה-**if** שלו. בדומה, הערה המתארת משמעות של **אי-קיום** תנאי מבהירה בשפה ברורה וקריאה את הסיבה לניתוב מהלך הביצוע של משפט ה-**if** אל תחום ה-**else** שלו.

חשוב לתעד באופן הזה משפטי **if** שאינם מקוננים, כפי שעשינו בתוכניות שהוצגו בפרק עד כה. חשוב עוד יותר לתעד מבנים מקוננים.

הנה שתי דוגמאות למבנים מקוננים שונים: בכל אחד משני המבנים הבאים מחושב יחס (קטן מ, גדול מ, שווה ל) בין ערכי שני משתנים (var1 ו-var2).

מבנה I:

```
if (var1 > var2)
    Console.WriteLine("var1 > var2");
else
{
    if (var1 == var2)
        Console.WriteLine("var1 = var2");
    else
        Console.WriteLine("var1 < var2");
}
```

מבנה II:

```
if (var1 >= var2)
{
    if (var1 == var2)
            Console.WriteLine("var1 = var2");
    else
        Console.WriteLine("var1 > var2");
}
else
    Console.WriteLine("var1 < var2");
```

שני המבנים כוללים הוראה אחת והיא הוראה לביצוע-בתנאי. במבנה I תחום ה-**else** של הוראה זו מכיל הוראה אחת וגם היא הוראה לביצוע-בתנאי במבנה *אם... אחרת...* גם במבנה II תחום ה-**if** מכיל הוראה לביצוע-בתנאי במבנה *אם... אחרת...* .

כאמור, הקפדה על הזחות מתאימות למשפטי **if** בתוכנית בשפת C# מיועדת לשמירה על קריאותה. המהדר של שפת C# אינו מייחס חשיבות להזחות. הוא אינו משייך תחום **else** למשפט ה-**if** המתאים לו על פי ההזחות. השיוך נקבע על פי הכלל הבא, שיודגם מיד:

כלל השיוך של **else** ל-**if** מתאים:

else תמיד משויך ל-**if** אשר נמצא בתוכנית לפניו וקרוב אליו ביותר בתנאים הבאים:

1. ל-**if** זה לא משויך **else** אחר קרוב יותר.
2. **if** זה איננו כלול בתחום סוגריים מסולסלים אחר שמסתיים עוד לפני ה-**else**.

כלל זה הוא מורכב למדי. הנה כמה דוגמאות שיסייעו בהבנתו:

- במבנה I שלעיל כל **else** משויך ל-**if** שנמצא לפניו וקרוב אליו ביותר.
- במבנה II שלעיל ה-**else** הראשון שייך ל-**if** אשר נמצא בתוכנית לפניו וקרוב אליו ביותר. לעומת זאת ה-**else** השני **אינו** משויך ל-**if** שנמצא לפניו וקרוב אליו ביותר, אלא משויך ל-if הראשון (החיצוני). זאת מכיוון שה-**if** הקרוב אליו ביותר כבר שויך ל-**else** אחר. לכן ה-**else** השני משויך ל-**if** הבא הקרוב ביותר, הלוא הוא ה-**if** הראשון.
- גם במבנה המקונן הבא ה-**else** איננו משויך ל-**if** שנמצא לפניו וקרוב ביותר אליו (ה-**if** השני). הפעם, מכיוון ש-**if** זה כלול בתחום סוגריים מסולסלים שמסתיים עוד לפני ה-**else**. לכן ה-**else** משויך ל-**if** הראשון (החיצוני) ולא ל-**if** השני (הפנימי).

```
if (var1 >= var2)
{
    if (var1 == var2)
        Console.WriteLine("var1 = var2");
}
else
    Console.WriteLine("var1 < var2");
```

שאלה 5.40

בכל אחד מהסעיפים הבאים נתון קטע תוכנית. כמו כן, בכל סעיף נתונות כמה אפשרויות לערכים התחלתיים של משתני הקטע. תארו את הפלטים של הקטעים הבאים, עבור כל אחד מהערכים ההתחלתיים של משתני הקטע.

א.

```
if (num > 0)
    Console.WriteLine("+");
else
{
    if (num == 0)
        Console.WriteLine("0");
    else
        Console.WriteLine("-");
}
```

ערך התחלתי של num	פלט
0	
10	
-10	

ב.

```
if (num1 > 0)
{
    if (num2 > 0)
        Console.WriteLine("+");
    else
        Console.WriteLine("+-");
}
else
    Console.WriteLine("-");
```

ערך התחלתי של num1	ערך התחלתי של num2	פלט
-1	0	
-5	5	
-1	5	

ג.

```
if (num1 < 0)
{
    if (num2 < 0)
        Console.WriteLine(num1 * num2);
}
else
{
    Console.WriteLine(num1);
    if (num2 < 0)
            Console.WriteLine(num1 + num2);
}
```

ערך התחלתי של num1	ערך התחלתי של num2	פלט
-1	1	
-1	-1	
0	0	

שאלה 5.41

הקלט בכל אחד מקטעי התוכניות הבאים הוא אות מן הא״ב האנגלי. מטרת כל אחד מקטעי התוכניות היא להציג כפלט הודעה אם האות הנקלטת היא האות N, אחת האותיות שקודמות ל-N בא״ב האנגלי או אחת האותיות שמופיעות אחרי N בא״ב האנגלי.

השלימו את קטעי התוכניות:

א. קטע תוכנית א

```
Console.Write("Enter a letter: ");
letter = char.Parse(Console.ReadLine());
if (letter == 'N')
        .
        .
        .
```

ב. קטע תוכנית ב

```
Console.Write("Enter a letter: ");
letter = char.Parse(Console.ReadLine());
if (letter != 'N')
        .
        .
        .
```

כפי שמראה התרגיל הבא, ניתן לעתים להמיר הוראה מקוננת לביצוע-בתנאי בהוראה לביצוע-בתנאי שאינה מקוננת אך כוללת תנאים מורכבים. עם זאת במקרים מסוימים השימוש בקינון הופך את התוכנית לקריאה ולברורה יותר.

שאלה 5.42

המירו את הפתרון לבעיה 7 `AnimalOlympics` לפתרון הכולל תנאים מורכבים, ואינו כולל קינון.

שאלה 5.43

קבוצת חייזרים ממאדים או מנוגה תגיע לבקר בכדור הארץ. יש לכתוב אלגוריתם המברך אותם, על פי כללי הטקס המסובכים הנהוגים בחלל.

שמה של קבוצת החייזרים ממאדים הוא תו כלשהו. לעומתה, שמה של קבוצת החייזרים מנוגה הוא מספר כלשהו. פתחו אלגוריתם הקולט מהיכן הגיעה הקבוצה: V (Venus) מנוגה ו-M (Mars) ממאדים. אם הקבוצה היא מנוגה, יש לקלוט את שמה (מספר) ואם הוא גדול מ-10 להציג את ההודעה "hello" ואחרת להציג את ההודעה "hi". אם הקבוצה היא ממאדים יש להציג את ההודעה "have a nice day".

ישמו את האלגוריתם בשפת C#.

האם ניתן לכתוב אלגוריתם לשאלה הנעזר בתנאים מורכבים ואינו משתמש בקינון של הוראות לביצוע-בתנאי, כפי שנעשה בתרגיל 5.42? אם כן, הראו כיצד. אם לא, הסבירו מדוע.

שאלה 5.44

נתון קטע התוכנית הבא, כאשר `let1`, `let2` ו-`let3` הם משתנים מטיפוס תווי השומרים אותיות מן הא״ב האנגלי:

```
if ((let1 == let2) || (let2 == let3) || (let1 == let3))
{
    if ((let1 == let2) && (let2 == let3))
        Console.WriteLine("1");
```

```
    else
        Console.WriteLine("2");
}
else
    Console.WriteLine("3");
```

א. הביאו דוגמת קלט שהפלט עבורה הוא 1, דוגמת קלט שהפלט עבורה הוא 2, ודוגמת קלט שהפלט עבורה הוא 3.

ב. צרפו תיאורי משמעות של קיום התנאים ושל אי-קיומם לתחום ה-**if** ולתחום ה-**else** שבמבנה המקונן.

ג. מהי מטרת קטע התוכנית?

שאלה 5.45

נתונים שני קטעי התוכניות הבאים, ובשניהם num הוא מטיפוס שלם:

א.

```
if(num < 0)
    Console.WriteLine("-1");

if (num == 0)
    Console.WriteLine("0");
else //1___________________
    Console.WriteLine("1");
```

ב.

```
if (num < 0)
    Console.WriteLine("-1");
else
{
    if (num == 0)
        Console.WriteLine("0");
    else //2___________________
        Console.WriteLine("1");
}
```

א. צרפו תיאורי משמעות לאי-קיום תנאי במקומות המסומנים

ב. האם שני קטעי התוכניות שקולים (כלומר, עבור כל קלט נתון יתקבל בשניהם אותו הפלט)? אם כן, הסבירו מדוע. אם לא, הביאו דוגמת קלט אשר עבורה יתקבלו פלטים שונים.

5.4 הוראת שרשרת לביצוע-בתנאי

לעתים נוח לכתוב הוראה לביצוע-בתנאי המתאימה לשרשרת של תנאים, שצריכים להיבדק זה אחר זה. בסעיף זה נתמקד בהוראות כאלו, במבנה *אם... אחרת אם ... אחרת...*

בעיה 8

מטרת הבעיה ופתרונה: הצגת הוראת שרשרת לביצוע-בתנאי במבנה *אם... אחרת אם ... אחרת...*

פתחו וישמו אלגוריתם שהקלט שלו הוא מספר המציין ציון במבחן והפלט שלו הוא הציון המילולי בתעודה, על פי המפתח הבא: כל ציון בין 90 ל-100 במבחן זוכה לציון A בתעודה. כל ציון בין 80 ל-89 במבחן זוכה לציון B בתעודה. כל ציון בין 70 ל-79 במבחן מקבל את הציון C בתעודה. כל ציון בין 60 ל-69 במבחן מקבל את הציון D בתעודה וכל ציון נמוך מ-60 במבחן מקבל את הציון F בתעודה.

פירוק הבעיה לתת-משימות

1. קליטת הציון במבחן
2. חישוב הציון המילולי בתעודה
3. הצגה כפלט של הודעה הכוללת את הציון המילולי בתעודה

בחירת משתנים

score – משתנה מטיפוס שלם לשמירת ציון המבחן

grade – משתנה מטיפוס תווי לשמירת הציון המילולי בתעודה

האלגוריתם

? בכתיבת האלגוריתם עלינו לנסח תנאי אשר יחשב את ציון התעודה המתאים לציון המבחן. מה יהיה התנאי הנחוץ?

הציון המילולי בתעודה מוכתב על ידי כמה תנאים:

1. ציון התעודה הוא A אם מתקיים התנאי: הציון במבחן גדול או שווה ל-90.
2. ציון התעודה הוא B אם מתקיים התנאי: הציון במבחן קטן מ-90 אבל גדול או שווה ל-80. זהו תנאי מורכב, שמשמעותו: תנאי 1 לא מתקיים וגם ציון המבחן גדול או שווה ל-80.
3. ציון התעודה הוא C אם מתקיים התנאי: הציון במבחן קטן מ-80 אבל גדול או שווה ל-70. זהו תנאי מורכב, שמשמעותו: תנאים 1 ו-2 לא מתקיימים וגם ציון המבחן גדול או שווה-70.
4. ציון התעודה הוא D אם מתקיים התנאי: הציון במבחן קטן מ-70 אבל גדול או שווה ל-60. זהו תנאי מורכב, שמשמעותו: תנאים 1, 2 ו-3 לא מתקיימים וגם ציון המבחן גדול או שווה ל-60.
5. ציון התעודה הוא D אם מתקיים התנאי: הציון במבחן קטן מ-60.

נוכל לבטא את ההתניה הזאת בשרשרת מורכבת של משפטי תנאי מקוננים, או בשימוש בתנאים מורכבים (ראו תרגילים בהמשך). לחילופין ניתן להיעזר בהוראה במבנה *אם... אחרת אם... אחרת...* שיוצרת מבנה ברור ופשוט למדי.

נציג אלגוריתם לפתרון הבעיה, שמשתמש בהוראה לביטוי שרשרת התנאים שניסחנו:

1. *קלוט את ציון המבחן ב-*score

2. *אם* $score \geq 90$

2.1. *השם ב-*grade *את התו* 'A'

3. *אחרת אם* $score \geq 80$

3.1. *השם ב-*grade *את התו* 'B'

4. *אחרת אם* $score \geq 70$

4.1. *השם ב-*grade *את התו* 'C'

5. *אחרת אם* $score \geq 60$

5.1. *השם ב-*grade *את התו* 'D'

6. *אחרת*

6.1. *השם ב-*grade *את התו* 'F'

יישום האלגוריתם

```
/*
קלט: ציון במבחן, בין 0 ל-100
```

```
פלט: ציון תעודה מילולי מתאים
*/
using System;
public class TermGrade
{
    public static void Main()
    {
         // הצהרה על משתנים בתוכנית
        int score;
        char grade;
        // קליטת המשתנים
1.      Console.Write("Enter test score: ");
2.      score = int.Parse(Console.ReadLine());
3.      if (score >= 90)
3.1.        grade = 'A';
4.      else if (score >= 80)
4.1.        grade = 'B';
5.      else if (score >= 70)
5.1.        grade = 'C';
6.      else if (score >= 60)
6.1.        grade = 'D';
7.      else
7.1.        grade = 'F';
8.      Console.WriteLine("Grade = {0}", grade);
    } // Main
}// class TermGrade
```

מעקב

נעקוב אחר מהלך ביצוע התוכנית TermGrade עבור הקלט 76:

מספר שורה	המשפט לביצוע	score	grade	score>=?	פלט
1	Console.Write("Enter test score: ");	?	?		Enter test score:
2	score = int.Parse(Console.ReadLine());	76	?		
3	**if** (score >= 90)	76	?	**false**	
4	**else if** (score >= 80)	76	?	**false**	
5	**else if** (score >= 70)	76	?	**true**	
5.1	grade = 'C';	76	C		
8	Console.WriteLine("Grade = {0}", grade);				Grade = C

סוף פתרון בעיה 8

בפתרון בעיה 8 מהלך הביצוע של האלגוריתם נקבע על פי שרשרת של תנאים. לצורך כך, השתמשנו במבנה *אם..אחרת אם...אחרת* של הוראות לביצוע-בתנאי.

הוראת שרשרת לביצוע-בתנאי כוללת סדרה של תנאים ושל הוראות לביצוע עבור כל תנאי. משמעות ההוראה היא כי התנאים נבדקים לפי הסדר עד שאחד מהם מתקיים. ברגע שתנאי מסוים מתקיים, מתבצעות ההוראות שהוגדרו עבורו, ושאר התנאים לא נבדקים.

הוראת שרשרת לביצוע-בתנאי (בצד ימין בכתיבה אלגוריתמית ובצד שמאל ב-C#):

אלגוריתמית	C#
אם <תנאי 1>	`if <תנאי 1>`
<קבוצת הוראות 1>	`{. . .}`
אחרת אם <תנאי 2>	`else if <תנאי 2>`
<קבוצת הוראות 2>	`{. . .}`
.	.
.	.
אחרת אם <תנאי k>	`else if <תנאי k>`
<קבוצת הוראות k>	`{. . .}`
אחרת	`else`
<קבוצת הוראות k+1>	`{. . .}`

שאלה 5.46

פתחו אלגוריתם המחשב את דרגת החוכמה של תוכים, הנקבעת באופן הבא: אם התוכי יודע לומר יותר מ-10 מילים הוא תוכי חכם מאוד. אם התוכי יודע לומר בין 6 ל-10 מילים הוא תוכי חכם. אם התוכי יודע לומר בין מילה אחת ל-5 מילים הוא תוכי ממוצע. אם התוכי לא יודע לומר אף מילה הוא תוכי שתקן.

האלגוריתם מקבל כקלט את מספר המילים השונות שתוכי יודע להגיד, ומציג הודעה המציינת את דרגת החוכמה של התוכי.

ישמו את האלגוריתם בשפת C#.

שאלה 5.47

פתחו אלגוריתם אשר הקלט שלו הוא מקדמים של משוואה ריבועית, a, b ו-c, והפלט הוא מספר הפתרונות הממשיים של המשוואה הריבועית (0 פתרונות, פתרון אחד או 2 פתרונות) והפתרונות עצמם. ישמו את האלגוריתם בשפת C#.

להזכירכם, הנה הנוסחה לחישוב פתרונותיה של משוואה ריבועית: $X_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

שאלה 5.48

נתונים שני קטעי התוכניות הבאים אשר בכל אחד מהם num הוא מטיפוס שלם.

האם שני קטעי התוכנית שקולים (כלומר, עבור כל קלט נתון יתקבל בשניהם אותו הפלט)? אם כן, הסבירו מדוע. אם לא, הביאו דוגמת קלט אשר עבורה יתקבלו פלטים שונים.

א.

```
if (num < 0)
    Console.WriteLine("-1");
else if (num == 0)
    Console.WriteLine("0");
else
    Console.WriteLine("1");
```

ב.

```
if (num < 0)
    Console.WriteLine("-1");
else
{
    if (num == 0)
        Console.WriteLine("0");
    else
        Console.WriteLine("1");
}
```

במובן מסוים, המבנה ***אם... אחרת אם... אחרת*** אינו חיוני. כפי שמראים התרגילים הבאים ניתן להסתדר גם בלעדיו. אבל השימוש בו מקל לעתים על הכתיבה, ויכול להפוך את התוכנית לקריאה יותר.

שאלה 5.49

המירו את האלגוריתם שבפתרון בעיה 8, לאלגוריתם שמשתמש בהוראה לביצוע-בתנאי במבנה מקונן במקום בהוראת שרשרת לביצוע-בתנאי.

שאלה 5.50

המירו את האלגוריתם שבפתרון בעיה 8, לאלגוריתם שמשתמש בתנאים מורכבים במקום בהוראת שרשרת לביצוע-בתנאי.

5.5 הוראת בחירה

במקרים רבים ערכו של ביטוי מסוים קובע את המשך המשימה. כלומר יש לבצע תת-משימה שונה עבור כל ערך אפשרי לביטוי. בכתיבת אלגוריתם מתאים למשימה כזאת נוח להשתמש בהוראת בחירה שנכיר בסעיף זה.

בעיה 9

מטרת הבעיה ופתרונה: הצגת הוראת בחירה המיושמת ב-C# במשפט **switch**.

פתחו וישמו אלגוריתם אשר מדמה מחשבון פשוט מאוד. הקלט הוא מספר ממשי, אחריו אחד מסימני הפעולה +, -, * או /, ואחריו מספר ממשי נוסף. הפלט הוא תוצאת החישוב של הפעולה על שני המספרים הנתונים.

למשל, עבור הקלט: 3.4 + 1.1- יהיה הפלט 2.3 -1.1+3.4=2.3

עבור הקלט: 1.2 / 4.8 יהיה הפלט 4 4.8/1.2= 4

פירוק הבעיה לתת-משימות

1. קליטת נתוני הקלט
2. ביצוע החישוב
3. הצגת תוצאת החישוב כפלט.

בחירת משתנים

num1 – משתנה מטיפוס ממשי, לשמירת המספר הראשון הניתן כקלט.
num2 – משתנה מטיפוס ממשי, לשמירת המספר השני הניתן כקלט.
operator – משתנה מטיפוס תווי, לשמירת סימן הפעולה הניתן כקלט.
result – משתנה מטיפוס ממשי, לשמירת תוצאת החישוב.

האלגוריתם

? התת-משימה המשמעותית היא התת-משימה השנייה. בתת-משימה זו יש לבצע אחד מארבעה חישובים: חיבור, חיסור, כפל או חילוק, לפי סימן הפעולה הנקלט. כיצד ננסח זאת?

דרך אחת לניסוח החישוב היא באמצעות המבנה המקונן הבא:

אם סימן הפעולה הוא '+'
חבר את שני המספרים

אחרת
אם סימן הפעולה הוא '-'
חסר את המספר השני מן הראשון
אחרת
אם סימן הפעולה הוא '*'
הכפל את שני המספרים
אחרת
חלק את המספר הראשון בשני

דרך אחרת לניסוח החישוב היא באמצעות הוראת שרשרת:

אם סימן הפעולה הוא '+'
חבר את שני המספרים
אחרת אם סימן הפעולה הוא '-'
חסר את המספר השני מן הראשון
אחרת אם סימן הפעולה הוא '*'
הכפל את שני המספרים
אחרת
חלק את המספר הראשון בשני

במקרה זה, שני המבנים הם מסורבלים יחסית. נוח יותר לכתוב הוראה המפרטת חלופות שונות בהתאם לערכו של סימן הפעולה, ועבור כל חלופה מנוסחת הוראת פעולה מתאימה.

ננסח את הפירוט של החלופות השונות באמצעות הוראת הבחירה הבאה:

בצע עבור הערך המתאים לסימן הפעולה:
'+': חבר את שני המספרים
'-': חסר את המספר השני מן הראשון
'*': הכפל את שני המספרים
'/': חלק את המספר הראשון בשני

שימו ♥: הפעולות חיבור, חיסור וכפל ניתנות לביצוע עבור כל זוג מספרים ממשיים, אך בפעולת חילוק יש להיזהר – אין לחלק ב-0. אם המחלק הוא 0, אין לבצע חלוקה, אלא רק להציג כפלט הודעת שגיאה על ניסיון חלוקה ב-0. נכלול זאת באלגוריתם המלא.

יישום האלגוריתם

הוראת בחירה מיושמת ב-C# במשפט **switch**, כפי שניתן לראות בתוכנית המלאה ליישום האלגוריתם שכתבנו:

```
/*
    התוכנית מדמה מחשבון לפעולות +, -, * ו-/
*/
using System;
public class Calculator
{
    public static void Main()
    {
        // הצהרה על משתנים בתוכנית
        double num1; // operand 1
        double num2; // operand 2
```

```
        char operator;     // operator
        double result;
        // קליטת המשתנים
1.      Console.Write("Enter the first number: ");
2.      num1 = double.Parse(Console.ReadLine());
3.      Console.Write("Enter the operator: ");
4.      operator = char.Parse(Console.ReadLine());
5.      Console.Write("Enter the second number: ");
6.      num2 = double.Parse(Console.ReadLine());
7.      switch (operator)
        {
7.1.        case '+':
7.1.1.          result = num1 + num2;
7.1.2.          Console.WriteLine("{0} + {1} = {2}",num1, num2,
                                                        result);
                break;
7.2.        case '-':
7.2.1.          result = num1 - num2;
7.2.2.          Console.WriteLine("{0} - {1} = {2}",num1, num2,
                                                        result);
                break;
7.3.        case '*':
7.3.1.          result = num1 * num2;
7.3.2.          Console.WriteLine("{0} * {1} = {2}",num1, num2,
                                                        result);
                break;
7.4.        case '/':
7.4.1.          if (num2 != 0)
                {
7.4.1.1.            result = num1 / num2;
7.4.1.2.            Console.WriteLine("{0} / {1} = {2}",num1, num2,
                                                        result);
                }
7.4.2.          else
7.4.2.1.            Console.WriteLine("Division by 0");
                break;
7.5.        default: Console.WriteLine("Illegal operator");
                break;
        } // switch
    } // Main
} // class Calculator
```

אם כך, בהוראת בחירה בוחנים ערך של ביטוי (במקרה זה, הביטוי הפשוט operator), ומשווים אותו לחלופות השונות. ביישום בשפת C# מקדימה המילה **case** את החלופות השונות.

שימו ♥: ה-**case** המתאים לחלופה שנבחרה, מהווה למעשה את נקודת הכניסה למשפט ה-**switch**. ממנו מתחילות להתבצע **כל** ההוראות שבתוך המשפט, זו אחר זו.

בסיום כל **case** שמפורטות בו פעולות לביצוע עלינו להוסיף את ההוראה **break** אשר מורה על יציאה ממשפט ה-**switch** ועל סיום ביצועו. אם ברצוננו לבצע אותן פעולות עבור **case** שונים, ניתן לכתוב את ה-**case** השונים בזה אחר זה, ורק עבור האחרון לכתוב את הפעולות לביצוע, ואחריהן להוסיף את ההוראה **break**. תרגיל 5.52 מדגים זאת.

קבוצת ההוראות הצמודה למילה **default** מתאימה לטיפול בערכים שאינם מטופלים באף חלופה, כלומר כאשר ערכו של הביטוי אינו שווה לאף אחד מהערכים המפורטים במשפט. אין חובה לכלול אפשרות **default** במשפט ה-**switch**. במקרה כזה, אם אין התאמה בין ערך הביטוי לאף אחד מהערכים המפורטים במשפט לא מתבצע דבר. גם בסיום חלק ה-**default** עלינו להוסיף את ההוראה **break**.

המעקב

המעקב אחר ביצוע משפט **switch** דומה למעקב אחר ביצוע משפט **if**. כלומר בשורה המתאימה בטבלת המעקב נרשום את המשפט **switch case:** לביצוע, ובשורה שאחריה נרשום את המשפט הנבחר לביצוע. הנה טבלת המעקב אחר מהלך ביצוע התוכנית Calculator עבור הקלט 2 / 5 :

מספר שורה	המשפט לביצוע	num1	num2	operator	result	פלט
1	Console.Write("Enter the first number: ")	?	?	?	?	Enter the first number
2	num1 = **double**.Parse(Console.ReadLine());	5	?	?	?	
3	Console.Write("Enter the operator: ");	5	?	?	?	Enter the operator
4	op = **char**.Parse(Console.ReadLine());	5	?	'/'	?	
5	Console.Write("Enter the second number: ");	5	?	'/'	?	Enter the second number
6	num1 = **double**.Parse(Console.ReadLine());	5	2	'/'	?	
7	**switch case**: /	5	2	'/'	?	
7.4.1	**if** (num2 != 0)	5	2	'/'	?	
7.4.1.1	result = num1 / num2;	5	2	'/'	2.5	
7.4.1.2	Console.WriteLine("{0} / {1}={2}",num1, num2, result);	5	2	'/'	2.5	5/2=2.5

סוף פתרון בעיה 9

שאלה 5.51

בנו טבלת מעקב אחר ביצוע התוכנית Calculator עבור הקלט 5.3 * 0.

כאשר עלינו לנתב את מהלך הביצוע על פי בחירה באחת מכמה אפשרויות של ערכים בדידים מתאים יותר להשתמש ב**הוראת בחירה** מאשר בהוראת שרשרת לביצוע-בתנאי.

המבנה הכללי של הוראת בחירה הוא :

עבור הערך המתאים של ביטוי בצע:

ערך 1: הוראה-1

ערך 2: הוראה-2

...

ערך k: הוראה-k

בביצוע ההוראה מחושב תחילה ערכו של הביטוי ולפיו מבוצעת החלופה המתאימה (שיכולה להיות הוראה מורכבת בפני עצמה).

הוראת בחירה מיושמת ב-C# ב**משפט switch**

המבנה הכללי של משפט **switch** הוא:

```
switch (ביטוי)
{
    case ערך: משפט; break;
    case ערך: משפט; break;
    .
    .
    .
    default: משפט; break;
}
```

תפקיד **משפט ה-break** הוא לגרום לסיום ביצוע הוראת ה-**switch**.

תפקיד ה-**default** (ברירת מחדל) הוא לספק הוראות שיתבצעו במקרה שערכו של הביטוי אינו שווה לאף אחת מן החלופות. אין חובה לכלול טיפול ברירת מחדל.

שאלה 5.52

בקטע התוכנית הבא יש שימוש ב-**break** רק בחלק מהמקרים:

```
switch (month)
{
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        numDays = 31;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        numDays = 30;
        break;
    case 2:
        if ( ((year%4 == 0) && (year%100 != 0)) || (year%400 == 0) )
            numDays = 29;
        else
            numDays = 28;
        break;
    default:
        Console.WriteLine("Error, Acceptable values for months " +
                                                      "are 1-12");
        break;
} // switch
```

א. מהי מטרת קטע התוכנית?

ב. תנו דוגמה לקלט שעבורו יגיע ביצוע משפט ה-**switch** לחלופת ברירת המחדל.
ג. תנו דוגמה לקלט שעבורו ערכו של המשתנה numDays בתום ביצוע משפט ה-**switch** יהיה 28 ודוגמה לערך שעבורו ערכו של המשתנה numDays בתום ביצוע משפט ה-**switch** יהיה 29.

שאלה 5.53

פתחו וישמו אלגוריתם אשר הקלט שלו הוא שלושה מספרים שלמים המציינים יום, חודש ושנה של תאריך. הפלט הוא תיאור התאריך בצורה יותר ברורה על ידי הצגת שם החודש, במקום מספרו.
למשל, עבור הקלט 1995 10 12 יהיה הפלט 12 באוקטובר 1995.

שאלה 5.54

במבחן יש 10 שאלות אמריקאיות, והציון עבור כל שאלה הוא 10 נקודות או 0 נקודות. כלומר ציון המבחן יכול להיות רק אחד מאחד עשר הציונים 0, 10, 20, ..., 90, 100.
פתחו אלגוריתם שהקלט שלו הוא ציון במבחן (כמספר שלם), והפלט שלו הוא הציון המילולי המתאים לו: לציון 100 במבחן מתאים הציון A, ל-90 מתאים B, ל-80 מתאים C, ל-70 מתאים D, ולכל ציון 60 ומטה מתאים F.

שאלה 5.55

כתבו תוכנית בשפת C# שתגריל קלף מתוך חפיסת קלפים ותדפיס את צורתו ואת ערכו. ערך של קלף הוא מספר בין 1 ל-13, וצורה של קלף היא: לב, תלתן, מעוין או עלה.
הדרכה : הגרילו מספר בין 1 ל-13 ומספר בין 1 ל-4. לפי התוצאה הציגו את ערך הקלף ואת צורתו (1 – לב, 2 – תלתן, 3 – מעוין, 4 – עלה).

סיכום

בפרק זה ראינו כיצד להורות על מהלכי ביצוע שונים באלגוריתם על פי קיום תנאי או על פי אי-קיומו. הדבר נעשה באמצעות הוראה לביצוע-בתנאי.

הוראה לביצוע-בתנאי מורה על בחירה לפי תנאי בין ביצוע תת-משימה אחת (פשוטה או מורכבת) ובין ביצוע תת-משימה אחרת: *אם... אחרת...*, או מורה על בחירה אם לבצע או לא לבצע תת-משימה: *אם...*.

התנאי המופיע בהוראה לביצוע-בתנאי יכול להיות תנאי פשוט או **תנאי מורכב**. תנאי מורכב בנוי מצירוף של תנאים פשוטים ושל הקשרים *וגם* ו-*או*.

ביטוי בוליאני מבטא תנאי. אם התנאי שמייצג הביטוי הבוליאני מתקיים אז ערכו של הביטוי הבוליאני הוא **true** (אמת). אם התנאי שמייצג הביטוי הבוליאני אינו מתקיים אז ערכו של הביטוי הבוליאני הוא **false** (שקר).

כאשר ביטויים בוליאניים מקושרים בקשר *או* ומרכיבים ביטוי בוליאני חדש, ערכו **true** כאשר **לפחות** אחד מערכי הביטויים הבוליאניים המקושרים הוא **true**. כאשר הם מקושרים בקשר *וגם*, ערכו של הביטוי החדש הוא **true** רק כאשר ערכי **כל** הביטויים הבוליאניים המקושרים הם **true**.

כאשר יש לנתב את מהלך הביצוע של האלגוריתם לבחירה מבין שלוש אפשרויות או יותר, ניתן לעתים להשתמש בהוראות מורכבות לביצוע-בתנאי:

מבנה מקונן של הוראות לביצוע-בתנאי הוא הוראה לביצוע-בתנאי שאחת (או יותר) מהוראותיה הפנימיות היא הוראה לביצוע-בתנאי בעצמה (פשוטה או מורכבת).

הוראת שרשרת לביצוע-בתנאי מתאימה לשרשרת של תנאים, שצריכים להיבדק זה אחר זה. התנאים נבדקים לפי הסדר עד שאחד מהם מתקיים. ברגע שתנאי מסוים מתקיים, מתבצעות ההוראות שהוגדרו בתחומו, ושאר התנאים לא נבדקים.

ב**הוראת בחירה** בוחנים ערך של ביטוי, ומשווים אותו לרשימת ערכים אפשריים. לפי תוצאת ההשוואה מתבצעת קבוצת הוראות.

כאשר מופיעה באלגוריתם הוראה לביצוע-בתנאי יש לבדוק את מהלך האלגוריתם עבור **קלטים מייצגים**, כלומר, לפחות דוגמת קלט אחת שתביא לכך שהתנאי יתקיים ולפחות דוגמת קלט אחת שתביא לכך שהתנאי לא יתקיים.

תיאורי משמעות של קיום תנאים ושל אי-קיומם הם הערות המבהירות את התנאי. הם עוזרים לנו בקריאת תוכנית. יש לצרף תיאורים כאלה לתנאים שכדאי להבהיר את משמעותם.

סיכום מרכיבי שפת C# שנלמדו בפרק 5

הוראה לביצוע-בתנאי במבנה *אם... אחרת...* נכתבת בשפת C# במשפט **if** באופן הבא:

```
if (ביטוי בוליאני)
{
    ההוראות אשר מבוצעות אם התנאי מתקיים
}
else
{
    ההוראות אשר מבוצעות אם התנאי אינו מתקיים
}
```

הוראה לביצוע-בתנאי במבנה *אם...* נכתבת בשפת C# במשפט **if** באופן הבא:

```
if (ביטוי בוליאני)
{
    ההוראות אשר מבוצעות אם התנאי מתקיים
}
```

ליצירת ביטויים בוליאניים פשוטים ב-C# ניתן להשתמש בכל אחד מ**סימני ההשוואה** של השפה. ניתן להשתמש בסימני ההשוואה כדי להשוות ערכים מכל טיפוס שערכיו ניתנים להשוואה. ניתן להשוות בין תווים.

הקשר *או* מסומן ב-C# בסימן הפעולה `||` **והקשר *וגם*** מסומן בסימן הפעולה `&&`. עדיפות הקשר `&&` גבוהה מעדיפות הקשר `||` כלומר `&&` קודם ל-`||` בסדר הפעולות. עם זאת מומלץ לסמן בבירור את טווח הפעולה של כל קשר בסוגריים, כדי ליצור ביטויים ברורים וקריאים.

כל אחת מההוראות בתוך משפט **if** ב-C# יכולה להיות בעצמה הוראה לביצוע-בתנאי מכל סוג שהוא. כך מתקבל **קינון** של הוראות לביצוע-בתנאי.

בהוראה מקוננת לביצוע-בתנאי, השיוך של **else** ל-**if** המתאים לו מתבצע על פי הכלל הבא:
else תמיד משויך ל-**if** אשר נמצא בתוכנית לפניו וקרוב אליו ביותר, בתנאים הבאים:

- ל-**if** זה לא משויך כבר **else** אחר קרוב יותר.
- **if** זה איננו כלול בתחום סוגריים מסולסלים אחר שמסתיים עוד לפני ה-**else**.

הוראת שרשרת לביצוע-בתנאי נכתבת בשפת C# באופן הבא:

```
if (ביטוי בוליאני 1)
{
    ההוראות אשר מבוצעות אם ערכו של ביטוי 1 הוא true
}
else if (ביטוי בוליאני 2)
{
    ההוראות אשר מבוצעות אם ערכו של ביטוי 2 הוא true
}
.
.
.
else
{
    ההוראות אשר מבוצעות אם ערכם של כל הביטויים הוא false
}
```

הוראת בחירה נכתבת בשפת C# במשפט **switch** באופן הבא:

```
switch (ביטוי)
{
    case ערך: <קבוצת הוראות> break;
    case ערך: <קבוצת הוראות> break;
    .
    .
    default: <קבוצת הוראות> break;
}
```

ערכו של הביטוי מושווה לכל אחד מהערכים המפורטים בחלופות השונות. **נקודת הכניסה** למשפט **switch** היא המשפט הצמוד לערך שעבורו הצליחה ההשוואה. מנקודת הכניסה מתבצעות כל ההוראות, עד להוראת **break**...

קבוצת ההוראות הצמודה ל-**default** (**ברירת המחדל**) מתבצעת אם ערכו של הביטוי המושווה אינו שווה לאף אחד מהערכים המפורטים. אין חובה לכלול טיפול ברירת מחדל. אם משפט **switch** אינו כולל חלופת ברירת מחדל, ובמהלך הביצוע ערכו של הביטוי המושווה אינו שווה לאף אחד מהערכים המפורטים, לא תבוצע אף הוראה.

שאלות נוספות

שאלות נוספות לסעיף 5.1

1. נתונים שני קטעי התוכניות הבאים, אשר המשתנים בהם הם מטיפוס שלם:

קטע תוכנית 1:

```
if (num < 0)
    num = -num;
Console.WriteLine(num);
```

קטע תוכנית 2:

```
if (num1 > num2 )
    diff = num1 - num2;
else
    diff = num2 - num1;
Console.WriteLine(diff);
```

א. עבור כל קטע תוכנית, בחרו שתי דוגמאות קלט כך שעבור האחת יתקיים התנאי המופיע בקטע ועבור השנייה לא יתקיים התנאי.

ב. מהי המטרה של כל קטע תוכנית?

ג. עבור כל קטע תוכנית, כתבו קטע תוכנית אחר ללא משפט **if**, המשיג את אותה המטרה.

2. התעריף לתשלום עבור צריכת גז ביתי לחימום הוא: 10 שקלים עבור כל אחד מחמשת הליטרים הראשונים הנצרכים ו-7 שקלים עבור כל ליטר נוסף.
נתון קטע תוכנית אשר הקלט שלו הוא כמות ליטרים שנצרכה הנתונה כמספר ממשי, והפלט שלו הוא התשלום הכולל והתשלום הממוצע עבור ליטר נצרך.
משתני קטע התוכנית הם מטיפוס ממשי: consumption מייצג את כמות הליטרים שנצרכה, payment מייצג את התשלום הכולל ו-average מייצג את התשלום הממוצע עבור ליטר נצרך.

```
if (consumption <= 5)
{
    השלימו
}
else
{
    השלימו
}
```

3. אחד השימושים של מחשב הוא הצפנת הודעות. דרך אחת להצפנת הודעות היא באמצעות החלפת תווים בתווים אחרים, ובאמצעות "ריפוד" בתווים נוספים.
פתחו וישמו אלגוריתם אשר הקלט שלו הוא זוג אותיות שונות מהא"ב האנגלי, והפלט שלו הוא שלושה תווים לפי החוקיות הבאה: אם אות הקלט הראשונה מופיעה בסדר האלף-בית האנגלי לפני אות הקלט השנייה אז הפלט הוא האות העוקבת באלף-בית לאות הקלט הראשונה, לאחריה הסימן '+' ולבסוף האות הקודמת באלף-בית לאות הקלט השנייה. אחרת הפלט הוא האות הקודמת באלף-בית לאות הקלט הראשונה, לאחריה הסימן '-' ולבסוף האות העוקבת באלף-בית לאות הקלט השנייה.
למשל, עבור הקלט ac יהיה הפלט b+b ועבור הקלט da יהיה הפלט c-b.

שאלות נוספות לסעיף 5.2

1. מהירות הנסיעה המותרת בכביש מהיר היא 55 קמ"ש לכל הפחות ו-100 קמ"ש לכל היותר. פתחו אלגוריתם אשר הקלט שלו הוא מהירות נסיעה של מכונית, והפלט שלו הוא הודעה אם נהג המכונית חרג מגבולות המהירות המותרת. ישמו את האלגוריתם בשפת C#.

2. פתחו וישמו אלגוריתם אשר הקלט שלו הוא מספר שלם חיובי קטן מ-100 והפלט שלו הוא המילה **בום** אם המספר הנתון מתחלק ב-7 או כולל את הספרה 7.

שאלות נוספות לסעיפים 5.3, 5.4 ו- 5.5

1. פתחו אלגוריתם הקולט תו. אם התו שווה ל-'m', האלגוריתם קורא מהקלט שני מספרים שלמים, ומציג כפלט את המספר הגדול מביניהם. אם תו הקלט שווה ל-'n', האלגוריתם קורא תו נוסף. אם התו הנוסף שווה ל-'o', האלגוריתם מציג את ההודעה `תחשוב חיובי`, אחרת הוא קולט מספר ומציג כפלט את ההודעה `אתה המספר שנקלט` ואת המספר עצמו. ישמו את האלגוריתם בשפת C#.

2. בחברת ההיי-טק "הייטק" יש 10 דרגות לעובדים. כל העובדים שהם מעל לדרגה 7 הם מנהלים, כל העובדים מעל לדרגה 4 הם ראשי צוותים, כל העובדים מעל לדרגה 2 הם עובדים קבועים וכל השאר הם סטודנטים. פתחו אלגוריתם (לשימוש מחלקת משאבי האנוש של החברה) הקולט את דרגת העובד ומציג כפלט את תיאורו של העובד (מנהל, ראש צוות, עובד קבוע או סטודנט). ישמו את האלגוריתם בשפת C#.

3. עזרו לזברה לדעת איזה יום היום. פתחו אלגוריתם הקולט מספר בין 1 ל-7 ומציג את השם של היום המתאים (Sunday, Monday...). ישמו את האלגוריתם בשפת C#.

שאלות מסכמות לפרק 5

1. בכל המשפטים שלהלן x הוא משתנה מטיפוס ממשי. כתבו עבור כל זוג ממשפטי ה-**if** הנתונים משפט **if** יחיד שקול, כלומר שביצועו שקול לביצוע שני משפטי ה-**if**, בזה אחר זה:

א.
```
if (x <= 9)
    x = x - 1;
if (x > 9)
    x = x + 1;
```

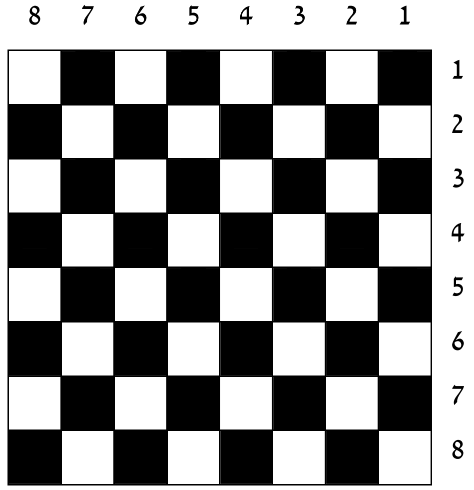
ב.
```
if (x >= 5)
    x = x * 4;
if (x >= 20)
    x = x / 2;
```

ג.
```
if (x == -1)
    x = 1;
if (x > 0)
    x = -x;
```

ד.
```
if (x < -1)
    x = -x;
if (x > 1)
    x = -x;
```

2. על לוח שחמט מוצבים שני כלים בלבד – צריח ורץ

השורות בלוח ממוספרות מ-1 עד 8, וגם העמודות ממוספרות מ-1 עד 8:



צריח (Rook באנגלית) מאיים על כלי הנמצא עמו באותה שורה או באותה עמודה.

רץ (Bishop באנגלית) מאיים על כלי הנמצא עמו על אותו אלכסון.

במשתנים `rookRow` ו-`rookCol` שמורים מספרי השורה והעמודה שהצריח מוצב עליהן.

במשתנים `bishopRow` ו-`bishopCol` שמורים מספרי השורה והעמודה שהרץ מוצב עליהן.

א. השלימו את תיאור משמעות קיום התנאי במשפט **if** הבא:

```
if((rookRow + rookCol) % 2) == 1)
      //_____________________________________
```

```
    Console.WriteLine("The rook is on …");
```

ב. כתבו ביטוי בוליאני שערכו **true** אם הצריח מאיים על הרץ.

ג. הביטוי הבוליאני הבא אמור לבטא מצב לוח שהרץ מאיים על הצריח:

```
(bishopRow - rookRow) == (bishopCol - rookCol)
```

הביטוי כולל רק חלק מן המקרים האפשריים. מהם המקרים הנכללים בו? מהם המקרים שאינם נכללים בו?

הרחיבו את הביטוי כך שיכלול את כל המקרים האפשריים ורק אותם.

תבניות – פרק 5

פירוט מלא של התבניות ושל שאלות שבפתרונן יש שימוש בתבניות ניתן למצוא באתר הספר ברשת האינטרנט.

מציאת מקסימום ומינימום בסדרה

שם התבנית: **מציאת מקסימום בסדרה**
נקודת מוצא: שני ערכים במשתנים element1 ו-element2
מטרה: מציאת הערך הגדול ביותר מבין שני הערכים
אלגוריתם:
1. *השם ב-*max *את הערך של* element1
2. *אם* element2 > max
 2.1 *השם ב-*max *את הערך של* element2

שם התבנית: **מציאת מינימום בסדרה**
נקודת מוצא: שני ערכים במשתנים element1 ו-element2
מטרה: מציאת הערך הקטן ביותר בין שני הערכים
אלגוריתם:
1. *השם ב-*min *את הערך של* element1
2. *אם* element2 < min
 2.1 *השם ב-*min *את הערך של* element2

סידור ערכים בסדרה

שם התבנית: **סידור ערכים בסדר עולה בסדרה**
נקודת מוצא: שני ערכים במשתנים element1 ו-element2
מטרה: השמת הערך הקטן יותר ב-element1 והערך הגדול יותר ב-element2
אלגוריתם:
1. אם element1 > element2
1.1 החלף את ערכי element1 *ו-*element2

שם התבנית: **סידור ערכים בסדר יורד בסדרה**
נקודת מוצא: שני ערכים במשתנים element1 ו-element2
מטרה: השמת הערך הגדול יותר ב-element1 והערך הקטן יותר ב-element2
אלגוריתם:
1. אם element1 < element2
1.1 החלף את ערכי element1 *ו-*element2

ערכים עוקבים

שם התבנית: **ערכים עוקבים?**
נקודת מוצא: שני ערכים element1 ו-element2
מטרה: קביעת הערך **true** אם element2 עוקב ל-element1, וקביעת הערך **false** אם element2 אינו עוקב ל-element1
אלגוריתם (ביטוי בוליאני):
(האם) element1 + 1 = element2

זוגיות מספר

שם התבנית: **מספר זוגי?**
נקודת מוצא: מספר שלם num
מטרה: קביעת הערך **true** אם num זוגי וקביעת הערך **false** אם num אי-זוגי
אלגוריתם (ביטוי בוליאני):
(האם) *שארית החלוקה של* num *פריטים ל-2 קבוצות שווה ל-0*

שם התבנית: **מספר אי-זוגי?**
נקודת מוצא: מספר שלם num
מטרה: קביעת הערך **true** אם num אי-זוגי וקביעת הערך **false** אם num זוגי
אלגוריתם (ביטוי בוליאני):
(האם) *שארית החלוקה של* num *פריטים ל-2 קבוצות שווה ל-1*

מחלק של מספר

שם התבנית: **מחלק של?**
נקודת מוצא: שני מספרים שלמים num1 ו-num2
מטרה: קביעת הערך **true** אם num2 מחלק את num1 וקביעת הערך **false** אם num2 אינו מחלק את num1
אלגוריתם (ביטוי בוליאני):
(האם) *שארית החלוקה של* num1 *פריטים ל-*num2 *קבוצות שווה ל-0*