שיעור 11

אופרטורים

מהו אופרטור?

אופרטור הוא סימן המאפשר לבצע פעולות אריתמטיות, השוואתיות ולוגיות בין מספרים או מחרוזות. למשל סימן החיבור (פלוס), +, מאפשר לבצע חיבור של שני מספרים, סימן החיסור (מינוס) מאפשר לבצע חיסור של מספר אחד ממספר שני.

אופרטורים אריתמטיים

אופרטור אריתמטי הוא פעולה חשבונית, למשל ארבע פעולות חשבות (חיבור, חיסור, כפל, חילוק) הן סוגים אופרטורים אריתמטיים.

סוגי האופרטורים האריתמטיים

אך ישנם עוד סוגים של אופרטורים אריתמטיים. סיכום של סוגי האופרטורים האריתמטיים מובא בטבלה הבאה:

האופרטור	מה הוא עושה	דוגמא לשימוש באופרטור
+	חיבור שני מספרים	C = A + B
-	חיסור שני מספרים	C = B - A
*	הכפלת שני מספרים	C = A * B
/	חלוקת שני מספרים	C = B / A
\	חלוקת שני מספרים והחזרת תוצאה של מספר שלם	C = B \ A
Mod	חלוקת שני מספרים והחזרת השארית (האופרטור Mod נקרא גם מודולו)	C = A Mod B
^	העלאת מספר בחזקה	B = A ^ 3
&	חיבור שתי מחרוזות	SiteName = "Doctor " & "VB"

עכשיו נפרט קצת על כל אופרטור:

חיבור (+)

אופרטור החיבור, + (פלוס), משמש לשם חיבורם של שני מספרים. את התוצאה מפעולת האופרטור נציב בדרך כלל למשתנה, לדוגמא:

```
Dim Result As Integer
Result = 200 + 300
```

בקוד הזה, הצהרנו על משתנה מסוג מספר שלם, בשם Result. ואחר כך הצבנו למשתנה את תוצאת החיבור של שני המספרים 200 ו-300. כעת ערכו של המשתנה Result יהיה תוצאת החיבור בין שני המספרים, כלומר 500.

דוגמא נוספת:

```
1 Dim Result As Integer
2 Dim NumberA As Integer
3 Dim NumberB As Integer
4 NumberA = 100
5 NumberB = 350
6 Result = NumberA + NumberB + 50
```

הסבר הדוגמא:
בשורות 1-3 הצהרנו על שלושה משתנים מסוג Integer (מספרים שלמים בלבד).
בשורה 4 הצבנו למשתנה NumberA את הערך 100.
בשורה 5 הצבנו למשתנה NumberB את הערך 350.
בשורה 6 הצבנו למשתנה Result את הערך מחיבור ערכי המשתנים NumberA, NumberB ועוד 50,
לכן ערכו של המשתנה Result יהיה כעת 500 (100 + 350 + 50).

חיסור (-)

אופרטור החיסור, - (מינוס), משמש לשם הפחת שני מספרים, לדוגמא:

```
1 Dim Result As Integer
2 Dim NumberA As Integer
3 NumberA = 220
4 Result = NumberA –180
```

הסבר הדוגמא:
בשורות 1-2 הצהרנו על שני משתנים מסוג Integer (מספרים שלמים בלבד).
בשורה 3 הצבנו למשתנה NumberA את הערך 220
בשורה 4 הצבנו למשתנה Result את תוצאת החיסור של 180 מערכו של המשתנה NumberA.
לכן ערכו של המשתנה Result יהיה כעת: 220 - 180 = 40

הפיכת מספר לשלילי:
פעולת החיסור, אינה הפעולה היחידה שאפשר לעשות בעזרת האופרטור -. בעזרת האופרטור – ניתן להפוך גם מספר לשלילי, לדוגמא:

```
Dim Result As Integer
Result = -90
```

בדוגמא למעלה, הדבר די ברור שסימן, – ,לפני מספר אומר שהמספר שלילי, אבל הדוגמא הבאה תבהיר עוד דבר:

```
1 Dim Result As Integer
2 Dim NumberA As Integer
3 NumberA = 320
4 Result = -NumberA
```

בשורות 1-2 הצהרנו על שני משתנים מסוג Integer (מספרים שלמים בלבד).
בשורה 3, הצבנו למשתנה NumberA את הערך 320.
בשורה 4, הצבנו למשתנה Result את הערך של NumberA כאשר לפניו שמנו את אופרטור החיסור (-), הדבר יגרום להפיכת הערך שמוצב במשתנה NumberA. כלומר כעת ערכו של המשתנה Result יהיה 320-. כלומר בדוגמא הזאת ראינו איך סימן מינוס לפני משתנה הופך את ערכו לשלילי.

כפל (*)

אופרטור הכפל, *, משמש לשם הכפלתם של שני מספרים, לדוגמא:

```
1 Dim Result As Single
2 Dim NumberA As Single
3 NumberA = 55.25
4 Result = NumberA * 2
```

בשורות 1-2 הצהרנו על שני משתנים מסוג Single (מספרים כולל שברים עשרוניים).
בשורה 3, הצבנו את הערך 55.25 למשתנה NumberA.
בשורה 4, הצבנו את התוצאה מהכפלת הערך של NumberA ב-2, כלומר ערכו של המשתנה Result יהיה כעת 110.5.

חילוק (/)

אופרטור החילוק, /, משמש לשם חילוק שני מספרים, לדוגמא:

```
1 Dim Result As Single
2 Dim NumberA As Integer
3 NumberA = 155
4 Result = NumberA / 2
```

בשורות 1-2 הצהרנו על שני משתנים אחד מסוג Single ואחד מסוג Integer.
בשורה 3 הצבנו למשתנה NumberA את הערך 155.
בשורה 4 הצבנו למשתנה Result את תוצאת החילוק של הערך של משתנה NumberA ב-2, כלומר ערכו של המשתנה Result יהיה כעת 77.5.

קבלת התוצאה השלמה מחילוק (\)

אופרטור החילוק, \, משמש לשם חילוק שני מספרים והחזרת התוצאה השלמה מהחילוק בלבד (בשונה מאופרטור החילוק, /, המחזיר את תוצאת החילוק, כולל השבר העשרוני).
לדוגמא:

```
1 Dim Result As Single
2 Dim NumberA As Integer
3 NumberA = 225
4 Result = NumberA \ 4
```

בשורות 1-2 הצהרנו על שני משתנים ובשורה 3 הצבנו למשתנה NumberA את הערך 225.
בשורה 4 חילקנו את הערך של המשתנה NumberA ב-4. התוצאה של חילוק כזה היא 56.25, אבל האופרטור \, מחזיר רק את התוצאה השלמה של החילוק, ולכן למשתנה Result תוצב רק התוצאה השלמה מהחילוק, כלומר 56.
חשוב להדגיש כי התוצאה לא מעוגלת כלפי מעלה או מטה, אלא מוחזר רק החלק השלם מהתוצאה. למשל אם התוצאה היתה 85.9, התוצאה המוחזרת תהיה 85.

<u>הערה חשובה:</u>
כאשר מחשבים חילוק של שני מספרים, שאחד מהם או שניהם הם מספרים עשרוניים, אז המספר העשרוני לפני ביצוע החילוק לתוצאה שלמה, יעוגל, ורק אחר כך תבוצע פעולת החילוק.
לדוגמא:

```
1 Dim Result As Single
2 Dim NumberA As Single
3 NumberA = 55.5
4 Result = NumberA \ 3.3
```

בשורות 1-2 הצהרנו על שני משתנים מסוג Single (מספרים כולל מספרים עשרוניים).
בשורה 3, הצבנו למשתנה NumberA את הערך 55.5.
בשורה 4 מה שמתבצע הוא:
1. קודם כל הערך של NumberA מתעגל מ-55.5 ל-56.
2. הערך של המספר המחלק 3.3 מתעגל ל-3.
3. התוצאה שתתקבל, תהיה המספר השלם המתקבל מחלוקת 56 ב-3, כלומר התוצאה תהיה 18.

מודולו - קבלת השארית מהחילוק (Mod)

אופרטור המודולו, Mod, משמש לשם חלוקת שני מספרים, והחזרת השארית מחילוקם. לדוגמא:

```
1 Dim Result As Single
2 Dim NumberA As Single
3 NumberA = 225
4 Result = NumberA Mod 4
```

בשורות 1-2 הצהרנו על שני משתנים, ובשורה 3 הצבנו למשתנה NumberA את הערך 225.
בשורה 4, הצבנו למשתנה Result את שארית החילוק של הערך של NumberA ב-4.
4 נכנס ב-225, 56 פעמים (4 * 56 = 224), ונשארת שארית של 1, ולכן הערך שיוצב ב-Result הוא 1.

חזקה (^)

אופרטור החזקה, ^, משמש לשם העלאת מספר בחזקה, לדוגמא:

```
1 Dim Result As Integer
2 Dim NumberA As Integer
3 NumberA = 4
4 Result = 4 ^ 2
```

בשורות 1-2 הצהרנו על שני משתנים, ובשורה 3, הצבנו למשתנה NumberA את הערך 4.
בשורה 4, הצבנו את הערך של 4 בחזקת 2 למשתנה Result, ולכן ערכו של המשתנה Result יהיה 16.

חיבור מחרוזות (&)

אופרטור החיבור, &, משמש לשם חיבור שתי מחרוזות, לדוגמא:

```
1 Dim Concatenation As String
2 Dim PartA As String
3 Dim PartB As String
4 PartA = "Visual "
5 PartB = "Basic"
6 Concatenation = PartA & PartB
```

בשורות 1-3 הצהרנו על 3 משתנים מסוג String (מחרוזת).
בשורה 4 הצבנו למשתנה PartA את המחרוזת " Visual".
בשורה 5 הצבנו למשתנה PartB את המחרוזת "Basic".
בשורה 6, הצבנו למשתנה Concatenation, את תוצאת החיבור של שתי המחרוזות, כלומר המחרוזת המאוכסנת כעת במשתנה Concatenation תהיה "Visual Basic".

אופרטורים השוואתיים

אופרטור השוואתי, משמש לשם ביצוע השוואה בין שני מספרים (או מחרוזות). אופרטור השוואתי יכול לקבוע האם המספרים שווים, שונים, או שמספר אחד גדול או קטן מהשני.
אופרטור השוואתי משווה בין המספרים או המחרוזות ומחזיר את הערך True אם ההשוואה מתקיימת, או את הערך False, כאשר ההשוואה אינה מתקיימת.

סוגי האופרטורים ההשוואתיים

הטבלה הבאה מסכמת את סוגי האופרטורים ההשוואתיים:

האופרטור	מה הוא עושה	דוגמא לשימוש באופרטור
=	בודק האם שני מספרים שווים.	Dim Alike As Boolean Alike = (45 = 50)
>	בודק האם מספר הראשון גדול מהשני.	Dim Bigger As Boolean Bigger = (100 > 90)
>=	בודק האם מספר הראשון גדול או שווה למספר שני.	Dim Result As Boolean Result = (100 >= 110)
<	בודק האם המספר הראשון קטן מהמספר השני.	Dim Result As Boolean Result = (100 < 110)
<=	בודק האם המספק הראשון קטן או שווה למספר השני.	Dim Result As Boolean Result = (100 <= 20)
<>	בודק האם שני מספרים שונים.	Dim Result As Boolean Result = (100 <> 20)
Is	בודק האם שני משתנים מיוחסים שווים (החזרת True) או שונים (החזרת False).	Dim Result As Boolean Dim obj As Object Set obj = Command1 Result = (obj Is Command1)
Like	בודק האם שתי מחרוזות שוות (True) או שונות (False).	Dim Alike As Boolean Dim PartA As String Dim PartB As String PartA = "Visual" PartB = "Basic" Alike = (PartA Like PartB)

האופרטורים: =, <>, <=, <, >=, >

אופרטורים אלו משווים בין מספרים וקובעים האם ההשוואה מתקיימת, לדוגמא:

```
1  Dim Alike As Boolean
2  Dim NumberA As Integer
3  Dim NumberB As Integer
4  NumberA = 100
5  NumberB = 222
6  Alike = (NumberA = NumberB)
7  MsgBox Alike
```

בשורות 1-3, הצהרנו על משתנה בוליאני בשם Alike ועל שני משתנים מסוג Integer.
בשורות 4-5 הצבנו למשתנה NumberA את הערך 100, ולמשתנה NumberB את הערך 222.
בשורה 6 בדקנו האם ערך המשתנה NumberA שווה לערך המשתנה NumberB, מכיוון שהערכים שבמשתנים לא שווים יוצב למשתנה Alike הערך False. אם ערכי המשתנים היו שווים היה מוצב למשתנה Alike הערך True.
בשורה 7 תציג תיבת הודעה קטנה את הערך של המשתנה Alike, כלומר את תוצאת ההשוואה

(על תיבת הודעה נלמד יותר בהמשך). התוצאה שתתקבל מהפעלת הקוד תהיה (כמובן שיש לשים את הקוד בתוך נוהל מסוים כמו לחיצה על כפתור וכדומה):

אם היינו מחליפים בקוד הנ"ל את שורה 6 לשורה הבאה:

```
Alike = (NumberA <= NumberB)
```

התוצאה המוצבת למשתנה Alike היתה True, מכיוון שכעת ההשוואה היא האם המשתנה NumberA קטן או שווה למשתנה NumberB. NumberA אומנם לא שווה למשתנה NumberB, אך הוא קטן ממנו ולכן מספיק שתנאי אחד יתקיים כדי שיוחזר הערך True.

השילוב של <=, =>, <>, הוא בעצם כמו של שני תנאים בהשוואה אחת.

בואו נראה דוגמא נוספת:

נחליף כעת את שורה 6 בקוד הנ"ל לשורה הבאה:

```
Alike = (NumberA <> NumberB)
```

שורת הקוד הזו בודקת האם הערך המוצב במשתנה NumberA שונה מהערך המוצב במשתנה NumberB. מכיוון שערכי המשתנים שונים, יוצב הערך True במשתנה Alike. אם ערכי המשתנים היו שווים היה מוצב למשתנה Alike הערך False.

הערה חשובה: למרות שמשתמשים באופרטורים אלו בדרך כלל להשוואת משתנים מספריים או מספרים, ניתן גם לבצע בעזרתם השוואת מחרוזות, לדוגמא:

```
1 Dim Alike As Boolean
2 Dim StringA As String
3 Dim StringB As String
4 StringA = “Doctor”
5 StringB = “VB”
6 Alike = (StringA < StringB)
7 MsgBox Alike
```

השוואת המחרוזות תתבצע על פי ערכי ANSI של האותיות שמכילה כל מחרוזת (לכל תו ב-Windows יש ערך ANSI שונה המייצג אותו), מכיוון שערך ה-ANSI של המחרוזת "Doctor" קטן יותר מזה של המחרוזת "VB" יוצב למשתנה Alike הערך True.

האופרטור Is

האופרטור Is בודק האם שני משתנים מיוחסים שווים, לדוגמא:

פתחו פרוייקט חדש בויזואל בייסיק, והוסיפו לטופס שנוצר כפתור פקודה בשם Command1.

הוסיפו את הקוד הבא לעורך הקוד של הטופס:

```
1 Private Sub Command1_Click()
2 Dim Result As Boolean
3 Dim obj As Object
4 Set obj = Command1
5 Result = (obj Is Command1)
6 MsgBox Result
7 End Sub
```

הסבר הקוד:

בשורה 1 הוספנו נוהל אירוע שיופעל כאשר המשתמש ילחץ על הכפתור Command1.

בשורה 2 הצהרנו על משתנה בוליאני בשם Result.

בשורה 3 הצהרנו על משתנה מיוחס בשם obj.

בשורה 4 קבענו שהמשתנה obj יהיה משתנה מיוחס לאובייקט כפתור הפקודה Command1.
בשורה 5 בדקנו האם המשתנה המיוחס obj שווה מבחינת יחוס לאובייקט Command1.
מכיוון שהמשתנה המיוחס obj מצביע לאובייקט Command1 התוצאה שתוצב למשתנה Result ותוצג בתיבת ההודעה בשורה 6, תהיה True.
אם היינו משנים את שורה 5 בקוד הנ"ל לשורה הבאה:

```
Result = (obj Is Form1)
```

אז התוצאה המתקבלת הייתה False כי המשתנה המיוחס obj מצביע ל-Command1 ולא ל-Form1.
אם היינו יוצרים עוד משתנה מיוחס שמצביע לאותו אובייקט ומשווים בינו לבין obj, אז היתה מוחזרת כמובן התוצאה True, לדוגמא הוסיפו את השורות הבאות במקום שורה 5 בקוד הנ"ל:

```
Dim ObjRef As Object
Set ObjRef = Command1
Result = (obj Is ObjRef)
```

ותראו כי השוואת שני המשתנים המיוחסים שמצביעים לאותו אובייקט תחזיר את הערך True.

האופרטור Like

האופרטור Like משווה בין שתי מחרוזות. לדוגמא:

```
Dim Alike As Boolean
Alike = (“Doctor” Like “VB”)
```

הדוגמא הזו תבדוק האם המחרוזת הראשונה דומה לשניה, ומכיוון שהמחרוזות אינן דומות, התוצאה שתוחזר תהיה False.

השוואה מתקדמת בעזרת Like:

אבל האופרטור Like מאפשר לבצע השוואות יותר מורכבות בין מחרוזות. למשל אם אתם מעונינים שהשוואה בין צורת הרבים של מילה (באנגלית, כמובן) לצורת היחיד שלה תחזיר True, אתם יכולים לבצע את ההשוואה הבאה:

```
1 Dim Alike As Boolean
2 Alike = ("Boxes" Like "Box*")
3 MsgBox Alike
```

הסבר:
התו * בא במקום כל מספר או תו ולכן בהשוואה בין "Boxes" ל-"Box*" יוחזר הערך True כי התו * יכול להחליף את ה-"es" ב-"Boxes".
חוץ מהתו *, ישנם עוד תווים שיכולים להחליף תו או תווים במחרוזת, הנה סיכום שלהם:

התו במחרוזת	מחליף...	דוגמא
*	מספר כלשהו של תווים (גם אפס)	Dim Alike As Boolean Alike = ("Visual" Like "Vi*al")
?	תו אחד בלבד	Dim Alike As Boolean Alike = ("Visual" Like "Vis?al")
#	ספרה אחת (בלבד) בין 0 ל-9	Dim Alike As Boolean Alike = ("VB 6.0" Like "VB #.0")

קדימות באופרטורים

בזמן שאתם משתמשים בביטוי, במספר אופרטורים ביחד, כל חלק בביטוי יתבצע לפי סדר קבוע מראש ולא לפי סדר כתיבתו. הדבר דומה מאוד לסדר הקדימות בארבע פעולות חשבון שכפל וחילוק קודמים לחיבור ולחיסור, אך צריך להתחשב כאן במספר נוסף של גורמים:

1. ביצוע אופרטורים אריתמטיים קודם לביצוע אופרטורים השוואתיים

לדוגמא:

```
1 Dim Result As Boolean
2 Result = (100 + 140 = 240)
3 MsgBox Result
```

בואו ננתח את הקוד:

בשורה 1 הצהרנו על משתנה בוליאני בשם Result.

בשורה 2 הצבנו למשתנה Result את הביטוי: 100 + 140 = 240

מכיוון שביצוע אופרטורים אריתמטיים קודם להשוואתיים, ויזואל בייסיק קודם תחשב את הסכום מחיבור 100 ו-140 (מכיוון שאופרטור החיבור הוא אופרטור אריתמטי), ורק אחר כך תבצע את ההשוואה ל-240. מכיוון שהסכום של 100 ו-140 שווה ל-240, הערך שיוצב למשתנה Result יהיה True, וזה יהיה גם הערך שיוצג בתיבת ההודעה בשורה 3.

2. קדימות באופרטורים אריתמטיים

אבל גם בתוך האופרטורים האריתמטיים יש סדרי קדימויות .

הטבלה הבאה מסכמת את סדר הקדימויות של האופרטורים האריתמטיים:

קדימות	האופרטור	שם האופרטור
1	^	חזקה
2	-	שלילה (הפיכת מספר לשלילי ולא חיסור)
3	* , /	כפל וחילוק
4	\	חילוק שלם (קבלת התוצאה השלמה מהחילוק)
5	Mod	מוֹדוּלוֹ – קבלת השארית מהחילוק
6	+, -	חיבור וחיסור
7	&	חיבור מחרוזות

דוגמא:

```
1 Dim Result As Integer
2 Result = (100 + 140 * 2)
3 MsgBox Result
```

בשורה 2 בדוגמא: קודם יבוצע הכפל של 140 ב-2, ורק אחר כך תחובר תוצאת המכפלה ל-100, מכיוון שאופרטור הכפל קודם לאופרטור החיבור.

3. קדימות באופרטורים השוואתיים

גם באופרטורים השוואתיים יש סדרי קדימויות, בדומה לאופרטורים אריתמטיים.

הטבלה הבאה מסכמת את סדר הקדימויות של האופרטורים ההשוואתיים:

קדימות	האופרטור	שם האופרטור
1	=	שוויון
2	<>	אי שוויון
3	<	קטן מ...
4	>	גדול מ...
5	<=	קטן או שווה ל...
6	>=	גדול או שווה ל...
7	Like, Is	שוויון/דמיון משתנים מיוחסים ומחרוזות

דוגמא:

```
1 Dim Result As Boolean
2 Result = (200 - 120 = 550 < 600)
3 MsgBox Result
```

בשורה 2, יתבצע קודם החיסור של 120 מ-200, כי אופרטורים אריתמטיים (כמו חיסור) קודמים לכל אופרטור השוואתי. אחר כך יבוצע השוויון, של 550 לתוצאת החיסור (80), כי אופרטור השוויון קודם לאופרטור ה-קטן מ... (סדר הקדימות באופרטורים השוואתיים).
מכיוון שהשוויון לא מתקיים, ערכה של ההשוואה יהיה False.
כעת תבוצע ההשוואה הבאה: האם התוצאה עד כה (False) קטנה מ-600. מכיוון שערכו המספרי של False הוא אפס, ההשוואה מתקיימת (כי 0<600) ולכן הערך שיוצב במשתנה Result יהיה True.

4. סוגריים:

אם ברצונך, למשל, שחיבורם של שני מספרים יתבצע לפני כפל במספר אחר, אז יש פשוט לבצע את פעולת החיבור בתוך סוגריים לדוגמא:

```
1 Dim Result As Integer
2 Result = 2 * (4 + 5)
```

בשורה 2, מכיוון שאופרטור החיבור נמצא בסוגריים, הוא יבוצע לפני הכפל, כלומר קודם יבוצע החיבור של 4 ועוד 5, ואחר כך תוצאת החיבור (9) תוכפל ב-2.
הדבר זהה למה שלמדתם בבית ספר בשיעורי חשבון.
ניתן כמובן להשתמש אף בסוגריים בתוך סוגריים, ובמקרה כזה תמיד יתבצע קודם האופרטור שנמצא בסוגריים הפנימיים יותר ואחר כך בסוגריים החיצוניים יותר וכך הלאה, לדוגמא:

```
1 Dim Result As Integer
2 Result = 2 * (4 * (5 - 3))
```

בדוגמא, קודם יתבצע החיסור בסוגריים הפנימיים (3 – 5), אחר כך יבוצע הכפל של תוצאת הסוגריים הפנימיים ב-4, ולבסוף יבוצע אופרטור הכפל של הכפלת התוצאה מהסוגריים ב-2, בדוגמא שלנו התוצאה תהיה 16.

הערה לסיום השיעור:

קיים סוג שלישי של אופרטורים, הנקרא "אופרטורים לוגיים", אליהם נתייחס בהמשך הלימוד.

שאלה לחזרה

איזו פעולה מבצע האופרטור Mod?

1. הוא לא מבצע כלום.
2. הוא גורם לשינוי הרזולוציה מ-640x480 ל-19220x14440.
3. הוא גורם לשינוי המצב רוח (Mod הוא קיצור של mood = מצב רוח).
4. הוא מחלק שני מספרים ומחזיר את השארית מהחילוק.