פעולות חשבון על טנזורים בעלי גודל שונה

כאשר עסקנו באופרטורים טנזוריים, ודנו בכך שהם הורמו לפעול איבר-איבר, הרצנו בין השאר את הדוגמה הבאה:

```
x = torch.tensor([1, 2, 3, 4])
print(x==1)
```

פלט:

```
tensor([ True, False, False, False])
```

על פניו, על הפקודה היה להשוות בין הטנזור x לבין הסקלר 1 ולהחזיר תשובה שלילית. פונקציונליות זו אינה שימושית במיוחד שכן התשובה להשוואה זו ברורה מאליה ולכן קודם להשוואה (איבר-איבר) הסקלר 1 עבר **שידור (broadcasting)**: מימדיו הושוו למימד הטנזור על ידי הכפלת הערך 1.
למעשה, הפעולה שהתבצעה מאחורי הקלעים שקולה לקטע הקוד הבא:

```
y = torch.tensor([1]*x.numel())
print(x, y, x==y, sep='\n')
```

פלט:

```
tensor([1, 2, 3, 4])
tensor([1, 1, 1, 1])
tensor([ True, False, False, False])
```

למעשה, פונקציונליות השידור מרחיקה לכת מעבר לשכפול בסיסי זה, ובאופן כללי כאשר ננסה להפעיל אופרטורים על טנזורים בעלי מימדים שונים, קודם לביצוע הפעולה אחד מהם (או שניהם יחדיו) יעבור שידור, מימדיו יגדלו והפעולה תבוצע רק לאחר ששני הטנזורים בעלי אותו ממד.

הדוגמאות הבאות מלמדות על הכללים של פעולת השידור, בהם נדון אחרי כן.

דוגמה 1:

```
A=torch.arange(9).reshape(3,3)
B=torch.arange(3)
print(A,B,A+B,sep='\n')
```

פלט:

```
tensor([[0, 1, 2],
        [3, 4, 5],
        [6, 7, 8]])
tensor([0, 1, 2])
tensor([[ 0,  2,  4],
        [ 3,  5,  7],
        [ 6,  8, 10]])
```

בדוגמה זו ניתן לראות שקודם לביצוע פעולת החיבור בין המטריצה A בעלת הגודל 3X3 לבין הטנזור החד ממדי B, B עבר שידור לאורך שורות A, ולמעשה בוצעה פעולת החיבור בין A לבין המטריצה הבאה:

```
B.expand_as(A)
```

פלט:

```
tensor([[0, 1, 2],
        [0, 1, 2],
        [0, 1, 2]])
```

שימו לב למתודה השימושית `expand_as()` המבצעת את השידור באופן גלוי ומאפשרת לנו לפרק כל פעולה הכוללת שידור לשלביה השונים ובחינתם בנפרד. כמו כן עליכם לדעת כי פקודה זו היא חסכונית בזכרון ולמעשה המטריצה המתקבלת היא מבט לתוך הנתונים של הטנזור `B` (כאשר העמודה הראשונה מצביעה אל אותו המקום בזכרון בו שמור הערך 0 וכו'). בהתאם, לאחר שימוש בפקודת שידור שכזו ולפני ביצוע פעולות המשנות את ערכיו של הטנזור במקומם – יש להשתמש במתודה `clone()` המשכפלת את הערכים למקום חדש בזכרון.

נשאלת השאלה מדוע שידור `B` התבצע דווקא לאורך השורות ולא לאורך העמודות. על פניו ייתכן היה שפעולת השידור תניב מטריצה אחרת, ובהתאם פעולת החיבור תניב תוצאה אחרת. ואכן, לו היינו מצהירים כי `B` הוא וקטור עמודה (מטריצה מסדר 3X1), תוצאת השידור הייתה משתנה בהתאם:

```
B.reshape(3,1).expand_as(A)
```

פלט:

```
tensor([[0, 0, 0],
        [1, 1, 1],
        [2, 2, 2]])
```

דוגמה 2:

```
A=torch.arange(3).reshape(3,1)
B=torch.arange(4).reshape(1,4)
print(A,B,A+B,sep='\n')
```

פלט:

```
tensor([[0],
        [1],
        [2]])
tensor([[0, 1, 2, 3]])
tensor([[0, 1, 2, 3],
        [1, 2, 3, 4],
        [2, 3, 4, 5]])
```

בדוגמה זו ניתן לראות ששתי המטריצות עברו שידור: עמודתה היחידה של `A` שוכפלה ארבע פעמים, בעוד ששורתה של `B` שוכפלה שלוש פעמים והתוצאה שהתקבלה היא סכום שתי המטריצות מסדר 3X4.

דוגמה 3:

```
A=torch.arange(6).reshape(3,2)
B=torch.arange(5).reshape(1,5)
print(A,B,sep='\n')
A+B
```

פלט:

```
tensor([[0, 1],
        [2, 3],
        [4, 5]])
tensor([[0, 1, 2, 3, 4]])
RuntimeError: The size of tensor a (2) must match the size of
tensor b (5) at non-singleton dimension 1
```

מדוגמה זו אנו למדים שלא כל זוג טנזורים מתאימים לשידור יחדיו. במקרה הנוכחי הסיבה לכך היא חוסר ההסכמה על גודל המימד השני: ל-A שתי עמודות בעוד של-B יש חמש.

משלושת הדוגמאות הנ"ל אנו לומדים את חוקי השידור הכלליים:

1. לפני ביצוע פעולה טנזורית איבר-איבר על שני טנזורים A, B בעלי מימדים שונים, יש לבדוק האם הם מתאימים לשידור:
   a. מתחילים מהמימד האחרון (הימני ביותר) של שני הטנזורים ובודקים ש:
      i. הם שווים זה לזה, או
      ii. אחד מהם שווה ל1.

   אם אחד מתנאים אלו אינו מתקיים – תתקבל שגיאה כגון זו שהתקבלה בדוגמה 3 לעיל.

   b. עוברים מימד אחד שמאלה בכל אחד מהטנזורים וחוזרים על הבדיקה לעיל.
   c. כאשר סיימנו לעבור על כל המימדים מימין לשמאל של לפחות אחד הטנזורים ולא התקבלה שגיאה, אפשר לעבור לשלב השידור.
2. בשלב השידור משכפלים את ערכי הטנזורים לפי החוקיות הבאה:
   a. אם אחד הטנזורים בעל מספר מימדים קטן יותר מהשני, מוסיפים בתחילתו מימדים מנוונים עד אשר מספר המימדים שווה.
   b. בכל מקרה שיש אי הסכמה בערך המימד – אחד הטנזורים הוא בעל מימד מנוון, אותו נשדר לאורך מימד זה, כגודל המימד של הטנזור השני.

נסיים פרק זה בדוגמה הממחישות את חוקי השידור.

דוגמה 4:

```
A=torch.arange(24).reshape(2,3,4)
B=torch.tensor([0,1,2,3])
C=B.expand_as(A)
D=A+B
print(A.size(),B.size(),C,D,sep='\n')
```

פלט:

```
torch.Size([2, 3, 4])
torch.Size([4])
tensor([[[0, 1, 2, 3],
         [0, 1, 2, 3],
         [0, 1, 2, 3]],

        [[0, 1, 2, 3],
         [0, 1, 2, 3],
         [0, 1, 2, 3]]])
tensor([[[ 0,  2,  4,  6],
         [ 4,  6,  8, 10],
         [ 8, 10, 12, 14]],

        [[12, 14, 16, 18],
         [16, 18, 20, 22],
         [20, 22, 24, 26]]])
```

בדוגמה זו ניתן לראות ראשית שהטנזורים A ו-B מתאימים לשידור, שכן המימד האחרון של A באותו גודל כמו המימד היחיד (ואחרון) של B. לאחר מכן B עובר הרחבה לטנזור תלת מימדי בגודל 1X1X4, ואחרי כן איבריו של B משודרים לאורך המימדים האחרים כך שפעולת החיבור מתבצעת למעשה בין

המטריצה A ו-C. לסיכום דוגמה זו, ודאו כי אתם מבינים שעבור כל הצבה חוקית של אינדקסים, החישוב הבא מניב ערך True.

```
C[i,j,k]==B[k]
```

שאלות לתרגול

1. נתונים טנזורים בעלי המימדים הבאים:

A	2X1X2
B	2X1X1X3
C	5X1
D	1X5
E	1
F	5X3

א. קבעו אילו זוגות של טנזורים מתאימים לשידור זה עם זה.

ב. לזוגות המתאימים לשידור – מהו מימד הטנזור המתקבל מהשידור?