

Math & Random

הספרייה המתמטית ומחלחת אקראי



יסודות מדעי המחשב

הספרייה המתמטית

Math

מספקת פונקציות מתמטיות וקבועים



יסודות מדעי המחשב

ביחידה זו נלמד:

Abs ✓

• מהי הספרייה Math

Pow ✓

• פעולות בספרייה Math

Sqrt ✓

Round ✓

• קבועים במחalkerah Math

Floor ✓

Min ✓

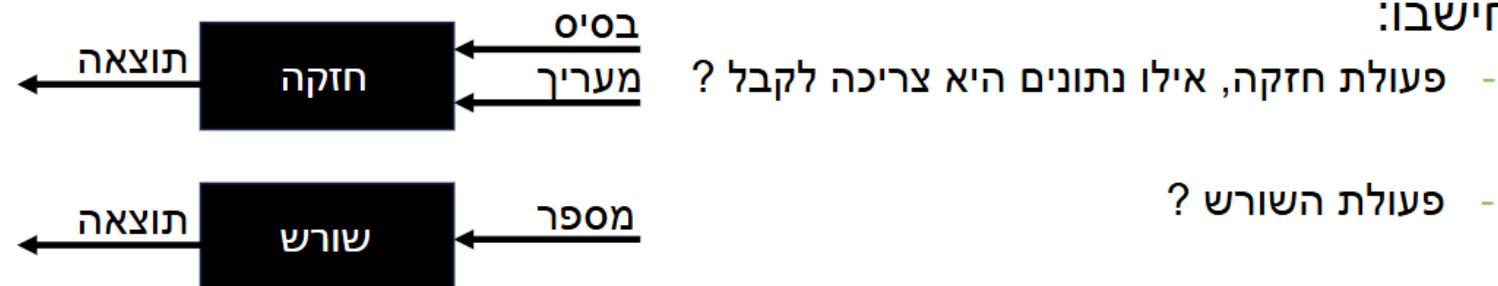
• הטיפוס Random להגרלת ערכים

Max ✓



מהי הספרייה ? Math

- זהה ספרייה מובנית המכילה אוסף פועלות מתמטיות.
- כל הפעולות מקובלות פרמטרים (נתונים), הנדרשים עבור הערכים שהן מתבקשות לחשב, ומחזירות את תוצאה החישוב.



הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס **שלם** או **עשרוני** – בהתאם לקלט.

```
static void AbsExamples()
{
}
```

הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס שלם או עשרוני – בהתאם לקלט.

```
static void AbsExamples()
{
    int res = Math.Abs(-5);
    Console.WriteLine (res);

}
```

הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס שלם או עשרוני – בהתאם לקלט.

```
static void AbsExamples()
{
    int res = Math.Abs(-5);
    Console.WriteLine (res);
}
```

5

הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס שלם או עשרוני – בהתאם לקלט.

```
static void AbsExamples()
{
    int res = Math.Abs(-5);
    Console.WriteLine (res);

    Console.WriteLine (Math.Abs(-8.3));
}
```

5

הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס שלם או עשרוני – בהתאם לקלט.

```
static void AbsExamples()
{
    int res = Math.Abs(-5);
    Console.WriteLine (res);

    Console.WriteLine (Math.Abs(-8.3));
```

5

8.3

הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס שלם או עשרוני – בהתאם לקלט.

```
static void AbsExamples()
{
    int res = Math.Abs(-5);
    Console.WriteLine (res);

    Console.WriteLine (Math.Abs(-8.3));
    Console.WriteLine (Math.Abs(7));
}
```

5

8.3

הפעולה **Abs**

- קיצור של המילה **Absolute** שמשמעותה **ערך מוחלט**.
- הפעולה מקבלת מספר (שלם או עשרוני) ומחזירה את הערך המוחלט שלו.
- הערך המוחזר יהיה מטיפוס **שלם** או **עשרוני** – בהתאם לקלט.

```
static void AbsExamples()
{
    int res = Math.Abs(-5);
    Console.WriteLine (res);           5

    Console.WriteLine (Math.Abs(-8.3));
    Console.WriteLine (Math.Abs(7));    8.3
}                                     7
```

הפעולה Pow

- הקייזור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעיר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8), יודפס הערך השלם

```
static void PowExamples()  
{  
}
```

הפעולה Pow

- הקיצור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעירך, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8.0), ידפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
}
```

הפעולה Pow

- הקיצור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעירך, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס **מספר עשרוני**.

- אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
}
```



הפעולה Pow

- הקיצור של המילה **Power** שימושוֹתָה חזקה.
- הפעולה מקבלת בסיס ומעיר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים /או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));
}
```

8

הפעולה Pow

- הקיצור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעיר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));
}
```

8
9

הפעולה Pow

- הקייזור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעירך, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים /או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.

- אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
}

}
```

8
9

הפעולה Pow

- הקייזור של המילה **Power** שימושוֹתָה חזקה.
- הפעולה מקבלת בסיס ומעירך, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
}
```

8
9
4

הפעולה Pow

- הकיצור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעירך, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.

- אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
    Console.WriteLine (Math.Pow(-2, 3));
}

}
```

8
9
4

הפעולה Pow

- הקייזור של המילה **Power** שימושו חזקה.
- הפעולה מקבלת בסיס ומעיר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
    Console.WriteLine (Math.Pow(-2, 3));
}

}
```

8
9
4
-8

הפעולה Pow

- הקייזור של המילה **Power** שמשמעותה חזקה.
- הפעולה מקבלת בסיס ומעירר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריר.
- הנתונים יכולים להיות שלמים, עשרוניים /או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.

- אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
    Console.WriteLine (Math.Pow(-2, 3));

    Console.WriteLine (Math.Pow(9, 0.5));
}


```

8

9

4

-8

הפעולה Pow

- הקייזור של המילה **Power** שימושוֹתָה חזקה.
- הפעולה מקבלת בסיס ומעירך, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הנתונים יכולים להיות שלמים, עשרוניים ו/או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.

- אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
    Console.WriteLine (Math.Pow(-2, 3));

    Console.WriteLine (Math.Pow(9, 0.5));
}


```

8
9
4
-8
3

הפעולה Pow

- הקיצור של המילה **Power** שימושוֹתָה חזקה.
- הפעולה מקבלת בסיס ומעירר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריר.
- הנתונים יכולים להיות שלמים, עשרוניים /או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.

- אם הערך המוחזר הואשלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
    Console.WriteLine (Math.Pow(-2, 3));

    Console.WriteLine (Math.Pow(9, 0.5));

    Console.WriteLine (Math.Pow(2, -1));
}
```

8
9

4
-8

3

הפעולה Pow

- הקיצור של המילה **Power** שימושוֹתָה חזקה.
- הפעולה מקבלת בסיס ומעיר, ומחזירה את תוצאה החישוב של הבסיס בחזקת המעריך.
- הنتונים יכולים להיות שלמים, עשרוניים /או שליליים.
- הערך המוחזר יהיה מטיפוס מספר עשרוני.
 - אם הערך המוחזר הוא שלם (לדוגמא 8.0), יודפס הערך השלם

```
static void PowExamples()
{
    Console.WriteLine (Math.Pow(2, 3));
    Console.WriteLine (Math.Pow(3, 2));

    Console.WriteLine (Math.Pow(-2, 2));
    Console.WriteLine (Math.Pow(-2, 3));

    Console.WriteLine (Math.Pow(9, 0.5));

    Console.WriteLine (Math.Pow(2, -1));
}
```

8
9
4
-8
3
0.5

הפעולה **Sqrt**

- קיצור של המילים **Square Root** שמשמעותו **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסוג עשרוני (אם הערךשלם כמו 4.0, יודפס השלים)

```
static void SqrtExamples()
{
}
```

הפעולה Sqrt

- קיצור של המילים **Square Root** שימושיתן **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסווג עשרוני (אם הערךשלם כמו 4.0, יודפס השלם)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));
}
```

הפעולה Sqrt

- קיצור של המילים **Square Root** שימושיתן **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסווג עשרוני (אם הערךשלם כמו 4.0, יודפס השלם)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));      4
}
```

הפעולה **Sqrt**

- קיצור של המילים **Square Root** שמשמעותו **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסוג עשרוני (אם הערךשלם כמו 4.0, יודפס השלם)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));
    Console.WriteLine(Math.Sqrt(16.5));
}
```

4

הפעולה **Sqrt**

- קיצור של המילים **Square Root** שמשמעותן **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חייבי בלבד).
- הערך המוחזר יהיה מסוג **עשרוני** (אם הערךשלם כמו 4.0, ידפס השלם)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));
    Console.WriteLine(Math.Sqrt(16.5));
}
```

```
4
4.06201920231798
```

הפעולה **Sqrt**

- קיצור של המילים **Square Root** שמשמעותו **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסווג עשרוני (אם הערךשלם כמו 4.0, יודפס השלים)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));          4
    Console.WriteLine(Math.Sqrt(16.5));         4.06201920231798
    Console.WriteLine(Math.Sqrt(15));
}
```

הפעולה **Sqrt**

- קיצור של המילים **Square Root** שמשמעותן **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסוג עשרוני (אם הערךשלם כמו 4.0, ידפס השלם)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));
    Console.WriteLine(Math.Sqrt(16.5));
    Console.WriteLine(Math.Sqrt(15));
}
```

```
4
4.06201920231798
3.87298334620742
```

הפעולה Sqrt

- קיצור של המילים **Square Root** המשמעותן **שורש ריבועי**.
- הפעולה מקבלת מספר ומחזירה את השורש הריבועי שלו.
- הפעולה יכולה לקבל כפרמטר ערךשלם או עשרוני (חיובי בלבד).
- הערך המוחזר יהיה מסווג עשרוני (אם הערךשלם כמו 4.0, יודפס השלם)

```
static void SqrtExamples()
{
    Console.WriteLine(Math.Sqrt(16));
    Console.WriteLine(Math.Sqrt(16.5));
    Console.WriteLine(Math.Sqrt(15));
}
```

```
4
4.06201920231798
3.87298334620742
```

- עבור שורש לא ריבועי, נשתמש בפעולת Pow כאשר פרמטר החזקה הוא שבר.
- למשל עבור $8^{1/3}$ המשמעות היא $8^{1/3}$, لكن נכתבת הפקודה כך:
`Console.WriteLine(Math.Pow(8, 1/3.0));`

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלים הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ولكن הוא יודפס כמספר שלם).

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
}
```

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלים הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
}
```

4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
}

}
```

4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ولكن הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
}

}
```

4
3

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));
}

}
```

4
3

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפו כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));
}
```

4
3
4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));
}
```

ערך החצוי מעוגל כלפי מעלה

4
3
4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלים הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));

    Console.WriteLine (Math.Round(-3.6));
}
```

ערך החצוי מעגל כלפי מעלה

4
3
4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));

    Console.WriteLine (Math.Round(-3.6));
}
```

ערך החצי מעוגל כלפי מעלה

4
3
4
-4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יחזיר ערך שלם וכך הוא ידפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));

    Console.WriteLine (Math.Round(-3.6));
    Console.WriteLine (Math.Round(-3.4));
}
```

ערך החצוי מעוגל כלפי מעלה

4
3
4
-4

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ولكن הוא ידפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));

    Console.WriteLine (Math.Round(-3.6));
    Console.WriteLine (Math.Round(-3.4));
}
```

ערך החצוי מעוגל כלפי מעלה

4
3
4

-4
-3

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יחזיר ערך שלם ולכון הוא ידפו כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));

    Console.WriteLine (Math.Round(-3.6));
    Console.WriteLine (Math.Round(-3.4));
    Console.WriteLine (Math.Round(-3.5));
}
```

ערך החצי מעוגל כלפי מעלה

4
3
4

-4
-3

הפעולה Round

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר.
- מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void RoundExamples()
{
    Console.WriteLine (Math.Round(3.6));
    Console.WriteLine (Math.Round(3.4));
    Console.WriteLine (Math.Round(3.5));

    Console.WriteLine (Math.Round(-3.6));
    Console.WriteLine (Math.Round(-3.4));
    Console.WriteLine (Math.Round(-3.5));
}
```

ערך החצוי מעוגל כלפי מעלה

4
3
4
-4
-3
-4

הפעולה `Floor`

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפו כמספר שלם).

```
static void FloorExamples()  
{
```

```
}
```

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשורי)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפו כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
}

}
```

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבורי במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכן הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
```

3

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא ידפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
}
```

3

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
}

}
```

3
3

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא ידפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));
}
```

```
3  
3
```

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));
}

}
```

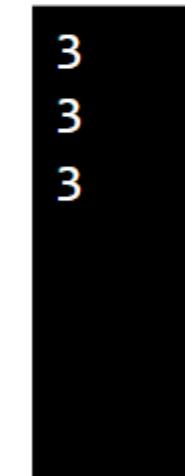
3
3
3

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));

    Console.WriteLine (Math.Floor(-3.6));
}
```

3
3
3


הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));

    Console.WriteLine (Math.Floor(-3.6));
}
```

3
3
3
-4

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));

    Console.WriteLine (Math.Floor(-3.6));
    Console.WriteLine (Math.Floor(-3.4));
}
```

3
3
3

-4

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשורי)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם ולכון הוא ידפו כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));

    Console.WriteLine (Math.Floor(-3.6));
    Console.WriteLine (Math.Floor(-3.4));
}
```

3
3
3
-4
-4

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלים הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסווג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסווג עשרוני (תמיד יוחזר ערך שלם ולכון הוא יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));

    Console.WriteLine (Math.Floor(-3.6));
    Console.WriteLine (Math.Floor(-3.4));
    Console.WriteLine (Math.Floor(-3.5));
}
```

3
3
3
-4
-4

הפעולה Floor

- מקבלת מספר ומעגלת אותו לשלם הקרוב ביותר כלפי מטה.
(המשמעות עבור במספרים חיוביים - קיצוץ הערך העשוני)
- הפעולה מקבלת כפרמטר מספר חיובי או שלילי מסוג שלם או עשרוני.
- הערך המוחזר יהיה מספר חיובי או שלילי מסוג עשרוני (תמיד יוחזר ערך שלם וכך יודפס כמספר שלם).

```
static void FloorExamples()
{
    Console.WriteLine (Math.Floor(3.6));
    Console.WriteLine (Math.Floor(3.4));
    Console.WriteLine (Math.Floor(3.5));

    Console.WriteLine (Math.Floor(-3.6));
    Console.WriteLine (Math.Floor(-3.4));
    Console.WriteLine (Math.Floor(-3.5));
}
```

3
3
3
-4
-4
-4

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיובים או שליליים) ומחזירות:

- Min את הערך הקטן מבין שני הערכים שהתקבלו.
- Max את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מחזירות:

- מספר שלם: אם שני הפרמטרים מティפו שלם.
- מספר עשרוני בכל מקרה אחר.

```
static void MinMaxExamples()
{
}
```

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיוביים או שליליים) ומחזירות:

- Min את הערך הקטן מבין שני הערכים שהתקבלו.
- Max את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מחזירות:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
}
```

- מספר שלם: אם שני הפרמטרים מティפוו שלם.
- מספר עשרוני בכל מקרה אחר.

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיוביים או שליליים) ומחזירות:
 - Min את הערך הקטן מבין שני הערכים שהתקבלו.
 - Max את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מחזירות:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
}
```

- מספר שלם: אם שני הפרמטרים מטיפוס שלם.
- מספר עשרוני בכל מקרה אחר.

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיוביים או שליליים) ומחזירות:

- Min את הערך הקטן מבין שני הערכים שהתקבלו.
- Max את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מחזירות:

- מספר שלם: אם שני הפרמטרים מטיפוס שלם.
- מספר עשרוני בכל מקרה אחר.

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
    Console.WriteLine(Math.Min(3, 5));
}
```

פעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיובים או שליליים) ומחירות:

- Min – את הערך הקטן מבין שני הערכים שהתקבלו.
- Max – את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מחירות:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
    Console.WriteLine(Math.Min(3, 5));
}
```

מספר שלם: אם שני הפרמטרים מטיפוס שלם.

3
3

מספר עשרוני בכל מקרה אחר.

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיובים או שליליים) ומחזירות:

- Min - את הערך הקטן מבין שני הערכים שהתקבלו.
- Max - את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מחזירות:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
    Console.WriteLine(Math.Min(3, 5));
    Console.WriteLine(Math.Min(-2, 4.5));
}
```

מספר שלם: אם שני הפרמטרים מטיפוס שלם.

3
3

מספר עשרוני בכל מקרה אחר.

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיוביים או שליליים) ומחזירות:

- Min את הערך הקטן מבין שני הערכים שהתקבלו.
- Max את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מוחזירות:

- מספר שלם: אם שני הפרמטרים מティפו שלם.

- מספר עשרוני בכל מקרה אחר.

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));           3
    Console.WriteLine(Math.Min(3, 5));           3
    Console.WriteLine(Math.Min(-2, 4.5));        -2
}
```

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיובים או שליליים) ומחזירות:

- Min – את הערך הקטן מבין שני הערכים שהתקבלו.
- Max – את הערך הגדול מבין שני הערכים שהתקבלו.

- הפעולות מוחזירות:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
    Console.WriteLine(Math.Min(3, 5));
    Console.WriteLine(Math.Min(-2, 4.5));

    Console.WriteLine(Math.Max(5, 3));
}
```

מספר שלם: אם שני הפרמטרים מティפו שלם.

3
3
-2

מספר עשרוני בכל מקרה אחר.

הפעולות Max ו Min

- הפעולות מקבלות כפרמטרים שני ערכים (שלמים או עשרוניים, חיובים או שליליים) ומחזירות:

- Min את הערך הקטן מבין שני הערכים שהתקבלו.
- Max את הערך הגדל מבין שני הערכים שהתקבלו.

- הפעולות מחזירות:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Min(5, 3));
    Console.WriteLine(Math.Min(3, 5));
    Console.WriteLine(Math.Min(-2, 4.5));

    Console.WriteLine(Math.Max(5, 3));
}
```

- מספר שלם: אם שני הפרמטרים מティפוס שלם.
- מספר עשרוני בכל מקרה אחר.

3	3
3	-2
5	

מעבר יותר מושני ערכים Max/Min

- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```

מעבר יותר מאשר Max/Min

- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

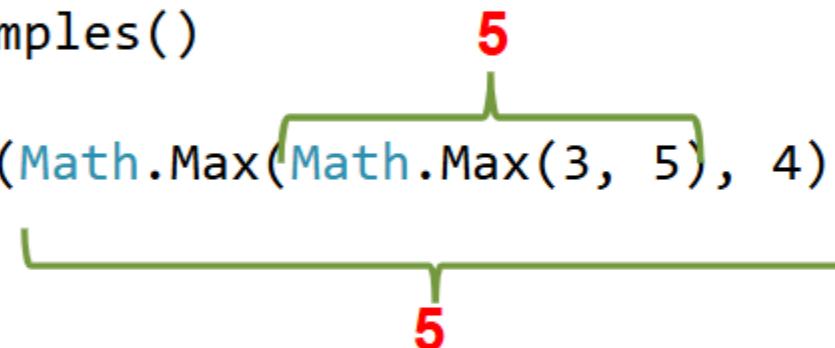
```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```

5

מעבר יותר משבבי ערכים Max/Min

- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

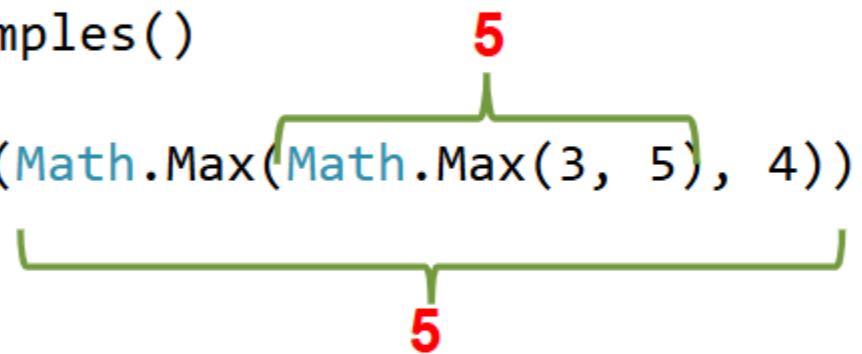
```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```



עבור יותר משלבי ערכים Max/Min

- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```



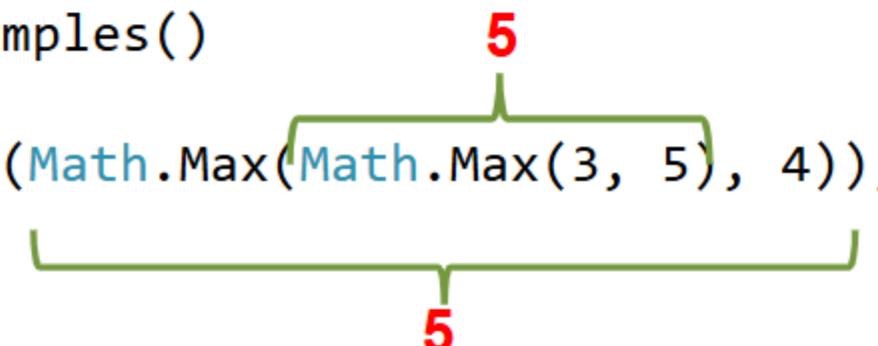
- על מנת למצוא מינימום או מקסימום בין 4 ערכים, יש להפעיל את הפעולה 3 פעמים:

```
static void MinMaxExamples()
{
}
```

מעבר יותר משלבי ערכים Max/Min

- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```



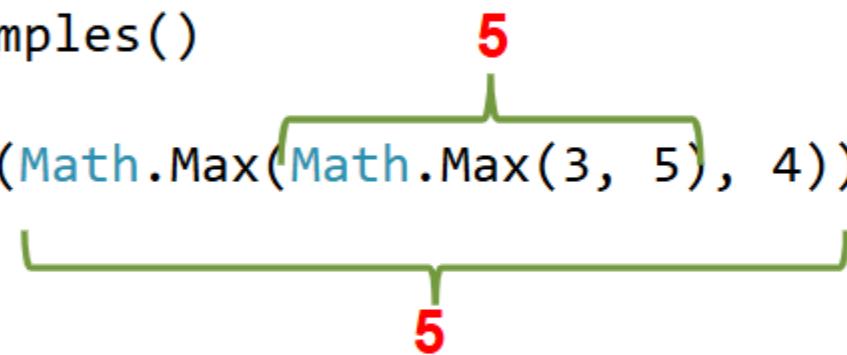
- על מנת למצוא מינימום או מקסימום בין 4 ערכים, יש להפעיל את הפעולה 3 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), Math.Max(4, 2)));
}
```

עבור יותר שניי ערכים Max/Min

- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```



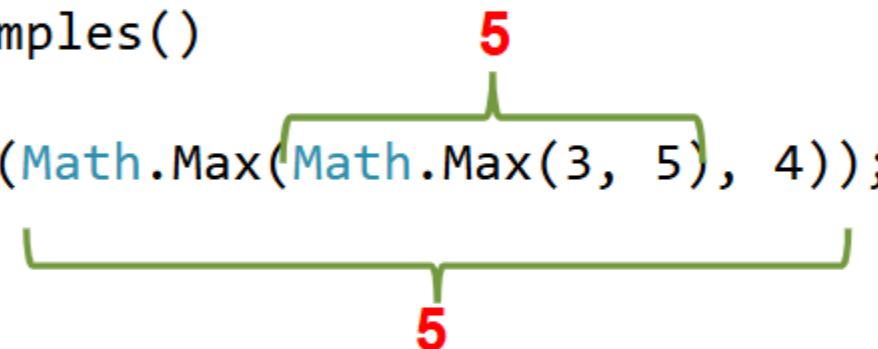
- על מנת למצוא מינימום או מקסימום בין 4 ערכים, יש להפעיל את הפעולה 3 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), Math.Max(4, 2)));
}
```

עבור יותר מושני ערכים Max/Min

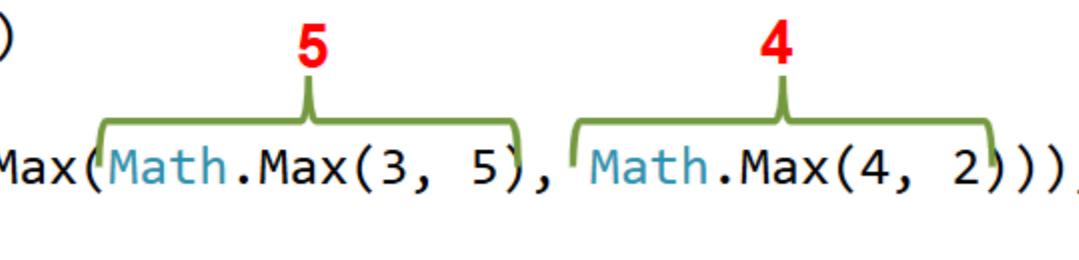
- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```



- על מנת למצוא מינימום או מקסימום בין 4 ערכים, יש להפעיל את הפעולה 3 פעמים:

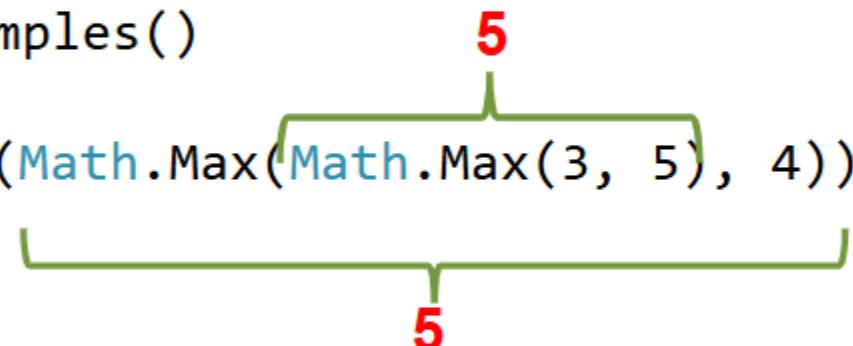
```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), Math.Max(4, 2)));
}
```



מעבר יותר משנה ערכים Max/Min

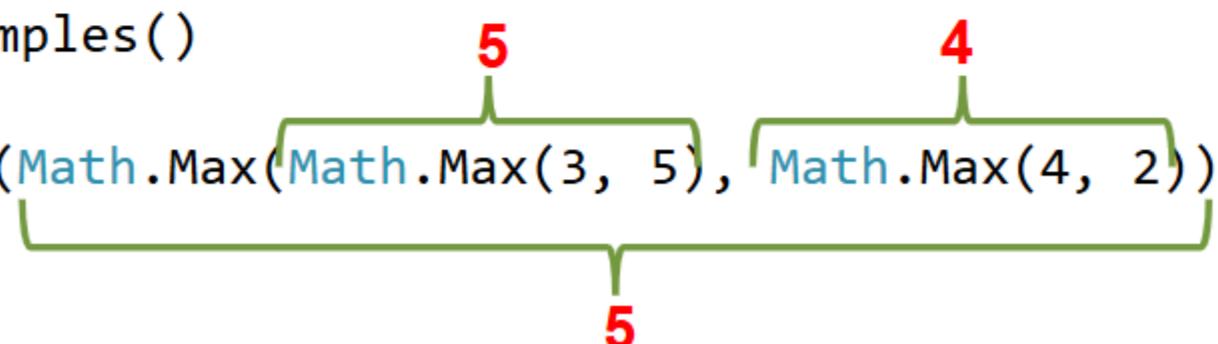
- על מנת למצוא מינימום או מקסימום בין 3 ערכים, יש להפעיל את הפעולה 2 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), 4));
}
```



- על מנת למצוא מינימום או מקסימום בין 4 ערכים, יש להפעיל את הפעולה 3 פעמים:

```
static void MinMaxExamples()
{
    Console.WriteLine(Math.Max(Math.Max(3, 5), Math.Max(4, 2)));
}
```



קבועים במחלקה Math

- המחלקה Math גם מגדרה מספר קבועים ידועים מעולם המתמטיקה ונitin להשתמש בהם.

```
static void ConstValuesInMath()
{
}
```

קבועים במחלקה Math

- המחלקה Math גם מגדרה מספר קבועים ידועים מעולם המתמטיקה וניתן לשימוש בהם.

```
static void ConstValuesInMath()
{
    Console.WriteLine(Math.PI);

}
```

קבועים במחלקה Math

- המחלקה Math גם מגדרה מספר קבועים ידועים מעולם המתמטיקה ונitin להשתמש בהם.

```
static void ConstValuesInMath()
{
    Console.WriteLine(Math.PI);
```

3.14159265358979

קבועים במחלקה Math

- המחלקה Math גם מגדרה מספר קבועים ידועים בעולם המתמטיקה ונitin להשתמש בהם.

```
static void ConstValuesInMath()
{
    Console.WriteLine(Math.PI);
    Console.WriteLine(Math.E);
}
```

3.14159265358979

קבועים במחלקה Math

- המחלקה Math גם מגדרה מספר קבועים ידועים מעולם המתמטיקה וניתן להשתמש בהם.

```
static void ConstValuesInMath()
{
    Console.WriteLine(Math.PI);
    Console.WriteLine(Math.E);
}
```

```
3.14159265358979
2.71828182845905
```

שימוש בקבועים מ-Math : חישוב היקף מעגל

```
static void CalcCirclePerimeter()
{
}
}
```

שימוש בקבועים מ-Math : חישוב היקף מעגל

```
static void CalcCirclePerimeter()
{
    int radius;
    double perimeter;

    Console.Write ("Enter a circle's radius: ");
    radius = int.Parse(Console.ReadLine ());

}
```

שימוש בקבועים מ-Math : חישוב היקף מעגל

```
static void CalcCirclePerimeter()
{
    int radius;
    double perimeter;

    Console.Write ("Enter a circle's radius: ");
    radius = int.Parse(Console.ReadLine());
}
```

Enter a circle's radius: 5

שימוש בקבועים מ-Math : חישוב היקף מעגל

```
static void CalcCirclePerimeter()
{
    int radius;
    double perimeter;

    Console.Write ("Enter a circle's radius: ");
    radius = int.Parse(Console.ReadLine());

    perimeter = 2 * Math.PI * radius;

    Console.WriteLine ("The circle's perimeter is " + perimeter);
}
```

Enter a circle's radius: 5

שימוש בקבועים מ-**Math** : חישוב היקף מעגל

```
static void CalcCirclePerimeter()
{
    int radius;
    double perimeter;

    Console.Write ("Enter a circle's radius: ");
    radius = int.Parse(Console.ReadLine());

    perimeter = 2 * Math.PI * radius;

    Console.WriteLine ("The circle's perimeter is " + perimeter);
}
```

```
Enter a circle's radius: 5
The circle's perimeter is 31.4159265358979
```

דוגמה מסכמת עם Math : משפט פיתגורס

עבור משולש ישר זווית שאורך ניצב אחד שלו הוא 6 ס"מ ואורך ניצב שני הוא 8 ס"מ.
יש לחשב מה אורך היתר במשולש (ניתן להשתמש במשפט פיתגורס).

יש לכתוב פעלת שתקלוט את אורך 2 הניצבים תחשב ותדפיס את אורך היתר.

```
static void CalcHypotenuse ()  
{  
  
}
```

דוגמה מסכמת עם Math : משפט פיתגורס

עבור משולש ישר זווית שאורך ניצב אחד שלו הוא 6 ס"מ ואורך ניצב שני הוא 8 ס"מ.
יש לחשב מה אורך היתר במשולש (ניתן להשתמש במשפט פיתגורס).

יש לכתוב פעליה שתקלוט את אורך 2 הניצבים תחשב ותדפיס את אורך היתר.

```
static void CalcHypotenuse ()  
{  
    double a = double.Parse(Console.ReadLine());  
    double b = double.Parse(Console.ReadLine());  
    double c = Math.Pow(a, 2);  
    c = c + Math.Pow(b, 2);  
    c = Math.Sqrt( c );  
    Console.WriteLine("The hypotenuse is " + c);  
}
```

דוגמה מסכמת עם Math : משפט פיתגורס

עבור משולש ישר זווית שאורק ניצב אחד שלו הוא 6 ס"מ ואורק ניצב שני הוא 8 ס"מ.
יש לחשב מה אורך היתר במשולש (ניתן להשתמש במשפט פיתגורס).

יש לכתוב פועלה שתקלוט את אורך 2 הניצבים תחשב ותדפיס את אורך היתר.

```
static void CalcHypotenuse ()  
{  
    double a = double.Parse(Console.ReadLine());  
    double b = double.Parse(Console.ReadLine());  
    double c = Math.Pow(a, 2);  
    c = c + Math.Pow(b, 2);  
    c = Math.Sqrt( c );  
    Console.WriteLine("The hypotenuse is " + c);  
}
```

```
4.0  
3.0  
The hypotenuse is 5
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בבדיקה מדויק

- אם המספר הוא 54.63887
- בדיק של 3 ספרות אחרי הנקודה קיבל 54.639
- בדיק של 2 ספרות אחרי הנקודה קיבל 54.64
- יש לכתוב פועלה שתקלוט מספר ממשי ומספר הספרות המדויק לאחר הנקודה העשרונית, תחשב ותדפיס את המספר המעווג.

רמז

$$5.1245218 * 10^3 = 5124.5218$$

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());    456.1234
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      456.1234
    int digits = int.Parse(Console.ReadLine());
}

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      456.1234
    int digits = int.Parse(Console.ReadLine());           2
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      456.1234
    int digits = int.Parse(Console.ReadLine());          2
    num = num * Math.Pow(10, digits);

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());           456.1234
    int digits = int.Parse(Console.ReadLine());                2
    num = num * Math.Pow(10, digits);                         456.1234*102 = 456.1234*100 = 45612.34
}
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקן מבוקש

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      456.1234
    int digits = int.Parse(Console.ReadLine());          2
    num = num * Math.Pow(10, digits);                  456.1234*102 = 456.1234*100 = 45612.34
    num = Math.Round(num);
}

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());           456.1234
    int digits = int.Parse(Console.ReadLine());                2
    num = num * Math.Pow(10, digits);                         456.1234*102 = 456.1234*100 = 45612.34
    num = Math.Round(num);                                    45612
}

}
```

דוגמה מוסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());           456.1234
    int digits = int.Parse(Console.ReadLine());                2
    num = num * Math.Pow(10, digits);                         456.1234*102 = 456.1234*100 = 45612.34
    num = Math.Round(num);                                    45612
    num = num / Math.Pow(10, digits);
}

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());
    int digits = int.Parse(Console.ReadLine());
    num = num * Math.Pow(10, digits);
    num = Math.Round(num);
    num = num / Math.Pow(10, digits);
}
```

456.1234
2
 $456.1234 \times 10^2 = 456.1234 \times 100 = 45612.34$
45612
 $45612 / 10^2 = 45612 / 100 = 456.12$

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());
    int digits = int.Parse(Console.ReadLine());
    num = num * Math.Pow(10, digits);
    num = Math.Round(num);
    num = num / Math.Pow(10, digits);
    Console.WriteLine(num);
}
```

456.1234
2
 $456.1234 \times 10^2 = 456.1234 \times 100 = 45612.34$
45612
 $45612 / 10^2 = 45612 / 100 = 456.12$

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());
    int digits = int.Parse(Console.ReadLine());
    num = num * Math.Pow(10, digits);
    num = Math.Round(num);
    num = num / Math.Pow(10, digits);
    Console.WriteLine(num);
}
```

456.1234
2
 $456.1234 \times 10^2 = 456.1234 \times 100 = 45612.34$
45612
 $45612 / 10^2 = 45612 / 100 = 456.12$

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());
    int digits = int.Parse(Console.ReadLine());
    num = num * Math.Pow(10, digits);
    num = Math.Round(num);
    num = num / Math.Pow(10, digits);
    Console.WriteLine(num);
}
```

456.1234
2
 $456.1234 \times 10^2 = 456.1234 \times 100 = 45612.34$
45612
 $45612 / 10^2 = 45612 / 100 = 456.12$

456.1234
2
456.12

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבוקש

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
}
```

דוגמה מוסכמת עם Math : עיגול מספר עשרוני בבדיקה מבקשת

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine());
}

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine()); 3
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine()); 3
    num = num * Math.Pow(10, digits);

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine()); 3
    num = num * Math.Pow(10, digits); 811.567812*103 = 811.567812*1000 = 811567.812
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine()); 3
    num = num * Math.Pow(10, digits); 811.567812*103 = 811.567812*1000 = 811567.812
    num = Math.Round(num);
}

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      811.567812
    int digits = int.Parse(Console.ReadLine());           3
    num = num * Math.Pow(10, digits);                   811.567812*103 = 811.567812*1000 = 811567.812
    num = Math.Round(num);                            811568
}

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine()); 3
    num = num * Math.Pow(10, digits); 811.567812*103 = 811.567812*1000 = 811567.812
    num = Math.Round(num); 811568
    num = num / Math.Pow(10, digits);

}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      811.567812
    int digits = int.Parse(Console.ReadLine());           3
    num = num * Math.Pow(10, digits);                   811.567812*103 = 811.567812*1000 = 811567.812
    num = Math.Round(num);                            811568
    num = num / Math.Pow(10, digits);                 811568/103 = 811568/1000 = 811.568
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקן מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine());      811.567812
    int digits = int.Parse(Console.ReadLine());           3
    num = num * Math.Pow(10, digits);                   811.567812*103 = 811.567812*1000 = 811567.812
    num = Math.Round(num);                            811568
    num = num / Math.Pow(10, digits);                 811568/103 = 811568/1000 = 811.568
    Console.WriteLine(num);
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void roundNum()
{
    double num = double.Parse(Console.ReadLine()); 811.567812
    int digits = int.Parse(Console.ReadLine()); 3
    num = num * Math.Pow(10, digits); 811.567812*103 = 811.567812*1000 = 811567.812
    num = Math.Round(num); 811568
    num = num / Math.Pow(10, digits); 811568/103 = 811568/1000 = 811.568
    Console.WriteLine(num);
}
```

811.567812
3
811.568

דוגמה מסכמת עם **Math** : עיגול מספר עשרוני בדיקת מבחן

```
static void RoundNum()
{
    double num = double.Parse(Console.ReadLine());
    int digits = int.Parse(Console.ReadLine());
    num = num * Math.Pow(10, digits);
    num = Math.Round(num);
    num = num / Math.Pow(10, digits);
    Console.WriteLine(num);
}
```

דוגמה מסכמת עם Math : עיגול מספר עשרוני בדיקת מבחן

```
static void RoundNum()
{
    double num = double.Parse(Console.ReadLine());
    int digits = int.Parse(Console.ReadLine());
    num = num * Math.Pow(10, digits);
    num = Math.Round(num);
    num = num / Math.Pow(10, digits);
    Console.WriteLine(num);
}
```

```
456.1234
2
456.12
811.567812
3
811.568
```

חלוקת אקראי

Random

אפשרת לקבל מספר אקראי (רנדומלי)
מתוך טווח מספרים

יסודות מדעי המחשב



מחלקה Random להגדרת ערכים

- כדי ליצור מספרים אקראיים (רנדומליים) ניצור עצם מטיפוס המחלקה `Random` `static Random r = new Random();`
- ואז "ນבקש" ממנו להציג לנו מספרים אקראיים ע"י שימוש בפעולת `Next`.
- פעולת `Next`מחזירה מספר אקראי מטיפוס מספר שלם (`int`).

`int x1 = r.Next();`
`int x2 = r.Next(5);`
`int x3 = r.Next(5,20);`

ניתן לקרוא ל פעולה `r.Next()` ב-3 צורות, לדוגמה:

חשוב:
כדי לוודא שאכן קיבל בכל הגרלה מספרים אקראי,
יש להגיד בק עצם אחד מטיפוס **Random**.

מחלקה Random : הגרלת ערכים

- כדי להגריל ערכים יש להגדיר עצם מטיפוס המחלקה Random.
- לעצם מטיפוס מחלקה זו יש פעולה Next שלא מקבלת פרמטרים.
 - ה פעולה מגרילה ערך חיובי כלשהו מסווג שלם.
 - כלומר בין 0 כולל למספר המקסימלי שנכנס במשתנה.

```
static Random r = new Random();  
  
static void RandNumbers()  
{  
    Console.WriteLine(r.Next() + " ");  
    Console.WriteLine(r.Next() + " ");  
    Console.WriteLine(r.Next() + " ");  
}
```

הגרלת ערך מספר שלם בטוחה 0
לערך המקסימלי האפשרי

1079670909
411725898
402634602

מחלקה Random : הגרלת ערכים חיוביים

- כדי להגריל ערכים יש להגדיר עצם מטיפוס המחלקה `Random`.
- כדי לקבל ערכים בטוחה יותר קטן, משתמש בפעולה `Next` המקבלת פרמטר אחד (`num`).
הפעולה מגירה ערך x מסווג שלם בטוחה $\text{num} < x \leq 0$.
כלומר בין 0 כולל ל-`num` לא כולל.

```
static Random r = new Random();
```

```
static void RandPositiveNumbers()
{
    Console.Write (r.Next(5) + " ");
    Console.WriteLine();
}
```

הגרלת ערך מספר שלם בטוחה בין 0 ל-4

1 4 3 0 1

2 3 2 2 3

מחלקה Random : הגרלת ערכים בטווח מסוים

- לטיפוס Random ישנה גרסה נוספת לפעולה Next, המקבלת 2 פרמטרים:
 - מספר שלם שהוא הערך הנמוך בטווח
 - מספר שלם שהוא הגודל ביותר בטווח לא כולל.

```
static Random r = new Random();
```

```
static void RandNumbersInRange()
{
    Console.Write (r.Next(10, 15) + " ");
    Console.WriteLine();
}
```

הגרלת ערך שלם בין 10 ל-15 (לא כולל)

11 14 13 10 12

14 13 12 12 13



הגרלת ערכי קובייה

- במקרה זה נגריל ערך אחד מבין 6 ערכים אפשריים :

```
static Random r = new Random();  
  
static void RandDiceNumbers()  
{  
    Console.WriteLine ("Rand values from dice:");  
    Console.Write (r.Next(1, 7) + " ");  
    Console.Write (r.Next(1, 7) + " ");  
    Console.Write (r.Next(1, 7) + " ");  
    Console.Write (r.Next(6)+1 + " ");  
    Console.Write (r.Next(6)+1 + " ");  
    Console.WriteLine();  
}
```

(0-5) +1

אפשר גם לשלב

Rand values from dice:
4 4 1 6 1

Rand values from dice:
4 3 4 6 4

הגרלת אותיות קטנות

- נוכל גם להגריל ערך ASCII אקראי עבור אות :

```
static Random r = new Random();  
  
static void RandLetters()  
{  
    char letter = (char)r.Next('a', 'z'+1);  
    Console.Write ((char)letter + " ");  
  
    Console.Write ((char)r.Next('a', 'z'+1) + " ");  
    Console.Write ((char)r.Next('a', 'z'+1) + " ");  
    Console.Write ((char)r.Next('a', 'z'+1) + " ");  
    Console.Write ((char)r.Next('a', 'z'+1) + " ");  
    Console.WriteLine();  
}
```

יש להמיר את הערך המתקבל ל-char

d v a g x

n g r h p

Decimal	Char
96	'
97	'a'
98	'b'
99	'c'
100	'd'
101	'e'
102	'f'
103	'g'
104	'h'
105	'i'
106	'j'
107	'k'
108	'l'
109	'm'
110	'n'
111	'o'
112	'p'
113	'q'
114	'r'
115	's'
116	't'
117	'u'
118	'v'
119	'w'
120	'x'
121	'y'
122	'z'
123	'{'
124	' '
125	'}'
126	'"
127	

הגרלת אותיות קטנות

- נוכל גם להגריל ערך Ascii אקראי עבור אות :

```
static Random r = new Random();

static void RandLetters()
{
    char letter = (char)r.Next('a', 'z'+1);
    Console.Write ((char)letter + " ");

    Console.Write ((char)r.Next('a', 'z'+1) + " ");
    Console.Write ((char)r.Next('a', 'z'+1) + " ");
    Console.Write ((char)r.Next('a', 'z'+1) + " ");
    Console.Write ((char)r.Next('a', 'z'+1) + " ");
    Console.WriteLine();
}
```

יש להמיר את הערך המתקבל ל-char

d v a g x

n g r h p

מחלקה Random : הגרלת ערכים ממשיים

- לטיפוס Random ישנה פעולה נוספת `NextDouble`, המגרילה מספרים ממשיים (לא שלמים).
- הגרלת מספרים ממשיים (עשרוניים) בין 0 (כולל) לבין 1 (לא כולל).
 - הפעולה אינה מקבלת פרמטרים.
 - הערך יוגרך תמיד בין 0 כולל לבין 1, לא כולל.
 - הערך המוחזר יהיה מסוג `מספר ממשי` (`double`).

```
static Random r = new Random();
```

```
static void RandDoubles()
{
    Console.WriteLine (r.NextDouble());
    Console.WriteLine (r.NextDouble());
    Console.WriteLine (r.NextDouble());
    Console.WriteLine (r.NextDouble());
    Console.WriteLine (r.NextDouble());
}
```

```
0.608843781803196
0.442635188550984
0.333353012955446
0.283628082500598
0.119824178107001
```

מחלקה Random : הגרלת ערכים ממשיים בתחום נתון

- הפעולה `NextDouble` של טיפוס `Random` ממחירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בתחום מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:

```
static Random r = new Random();
static void RandDoubles()
{
}
}
```

מחלקה Random : הגרלת ערכים ממשיים בתחום נתון

- הפעולה של טיפוס Random מחייבת תמיד ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בתחום מסוים (בין min ל-max) נגריל ונבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).

```
static Random r = new Random();
static void RandDoubles()
{
    double d = r.NextDouble();
}
```

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתקיים יהיה בטווח בין 0 להפרש `max-min`, לא כולל).

```
static Random r = new Random();
static void RandDoubles()
{
    double d = r.NextDouble();
    d = d * (max-min);
}
```

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתקיים יהיה בטווח בין 0 להפרש `max-min`, לא כולל).
 - נוסיף את הערך המינימלי.

```
static Random r = new Random();
static void RandDoubles()
{
```

```
    double d = r.NextDouble();
    d = d * (max-min);
    d = d + min;
```

```
}
```

מחלקה Random : הגרלת ערכים ממשיים בתחום נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בתחום מסוים (בין `min` ל-`max`) נגריל ונבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `chom` ל-`max` (הערך המתקיים יהיה בתחום בין 0 להפרש `max-min`, לא כולל).
 - נוסיף את הערך המינימלי.

```
static Random r = new Random();  
static void RandDoubles()  
{
```

לדוגמא:

הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

```
    double d = r.NextDouble();  
    d = d * (max-min);  
    d = d + min;  
  
}
```

מחלקה Random : הגרלת ערכים ממשיים בתחום נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בתחום מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתתקבל יהיה בתחום בין 0 להפרש `max-min`, לא כולל).

```
static Random r = new Random();  
static void RandDoubles()  
{  
    int min= int.parse(Console.ReadLine());  
    int max= int.parse(Console.ReadLine());  
  
    double d = r.NextDouble();  
    d = d * (max-min);  
    d = d + min;  
}
```

10
15

לדוגמא:
הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

מחלקה Random : הגרלת ערכים ממשיים בתחום נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בתחום מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתתקבל יהיה בתחום בין 0 להפרש `max-min`, לא כולל).
 - מוסיף את הערך המינימלי ונתקבל המינימלי.

```
static Random r = new Random();
static void RandDoubles()
{
    int min= int.parse(Console.ReadLine());
    int max= int.parse(Console.ReadLine());
    double d = r.NextDouble();
    d = d * (max-min);
    d = d + min;
}
```

10
15

לדוגמא:
הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתקיים יהיה בטווח בין 0 להפרש `max-min`, לא כולל).
 - נוסיף את הערך המינימלי
ונקבל מספר ממשי בין `min` ל-`max`.

```
static Random r = new Random();
static void RandDoubles()
{
    int min= int.parse(Console.ReadLine());
    int max= int.parse(Console.ReadLine());          10
    double d = r.NextDouble();           0 ... 1
    d = d * (max-min);
    d = d + min;
}
```

לדוגמא:

הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- פעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתקיים יהיה בטווח בין 0 להפרש `max-min`, לא כולל).
 - מוסיף את הערך המינימלי

```
static Random r = new Random();
static void RandDoubles()
{
    int min= int.parse(Console.ReadLine());
    int max= int.parse(Console.ReadLine());
    double d = r.NextDouble();      0 ... 1
    d = d * (max-min);
    d = d + min;
}
```

10
15

לדוגמא:
הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ונבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתתקבל יהיה בטווח בין 0 להפרש `max-min`, לא כולל).
 - נוסיף את הערך המינימלי ונקבל מספר ממשי בין `min` ל-`max`.

```
static Random r = new Random();  
static void RandDoubles()
```

```
{  
    int min= int.parse(Console.ReadLine());  
    int max= int.parse(Console.ReadLine());  
  
    double d = r.NextDouble();          0 ... 1  
    d = d * (max-min);                0 ... 5  
    d = d + min;  
  
}
```

לדוגמא:

הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתתקבל יהיה בטווח בין 0 להפרש `max-min`, לא כולל).
 - נוסיף את הערך המינימלי

```
static Random r = new Random();
static void RandDoubles()
{
    int min= int.parse(Console.ReadLine());
    int max= int.parse(Console.ReadLine());

    double d = r.NextDouble();      0 ... 1
    d = d * (max-min);           0 ... 5
    d = d + min;
}
```

10
15

לדוגמא:
הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- פעולה `NextDouble` של טיפוס `Random` ממחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתקיים יהיה בטווח בין 0 להפרש `max-min`, לא כולל).

```
static Random r = new Random();  
static void RandDoubles()
```

```
{  
    int min= int.parse(Console.ReadLine());  
    int max= int.parse(Console.ReadLine());
```

10
15

```
double d = r.NextDouble();      0 ... 1  
d = d * (max-min);           0 ... 5  
d = d + min;                 10... 15
```

לדוגמא:

הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

}

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחזירה **תמיד** ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ומבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתתקבל יהיה בטווח בין 0 להפרש `max-min`, לא כולל).
 - נוסיף את הערך המינימלי
ונקבל מספר ממשי בין `min` ל-`max`.

```
static Random r = new Random();
static void RandDoubles()
```

```
{  
    int min= int.parse(Console.ReadLine());  
    int max= int.parse(Console.ReadLine());  
  
    double d = r.NextDouble();  
    d = d * (max-min);  
    d = d + min;  
    Console.WriteLine(d);  
}
```

10
15

0 ... 1
0 ... 5
10... 15

לדוגמא:
הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

10.3376900871925

מחלקה Random : הגרלת ערכים ממשיים בטווח נתון

- הפעולה `NextDouble` של טיפוס `Random` מחייבת תמיד ערכים בין 0 ל-1 (לא כולל).
- כדי לקבל ערכים אקראיים בטווח מסוים (בין `min` ל-`max`) נגריל ונבצע חישוב:
 - נגריל מספר ממשי (הערך בין 0 כולל לבין 1, לא כולל).
 - נכפול אותו בהפרש בין `min` ל-`max` (הערך המתקיים יהיה בטווח בין 0 להפרש `max-min`, לא כולל).

```
static Random r = new Random();
static void RandDoubles()
```

```
{  
    int min= int.parse(Console.ReadLine());      10  
    int max= int.parse(Console.ReadLine());      15  
  
    double d = r.NextDouble();                  0 ... 1  
    d = d * (max-min);                      0 ... 5  
    d = d + min;                            10... 15  
    Console.WriteLine(d);  
}
```

לדוגמא:

הגרלת מספר ממשי בין 10 ל-15 (לפי קלט)

```
10.3376900871925  
14.9544938723345  
10.3676198773867  
11.5951304913476  
10.8844062550386
```

```
Random r = new Random();
```

הזמן	דוגמאות	טווח ערכים אפשרי
r.Next (10)	2, 7, 4, 9	0, 1, 2, 3 ...9
r.Next (-10)		שגיאה – פרמטר אחד מגריל טווח מ-0 עד למספר חיובי
r.Next (10, 50)	17, 33, 14	10, 11, 12, 13 ...49
r.Next (50, 10)		שגיאה! המספר השני קטן מהראשון
r.Next (-50, 10)	-46, 7, 0	-50, -49, -48 ... 9
r.Next (1)	0	0

```
Random r = new Random();
```

הזמן	דוגמאות	מה מקבל ?
r.Next(10)	3, 7, 4, 1	מספר
r.Next(10,100)	34, 56, 78	מספר דו-ספרתי
r.Next(49) * 2 + 1	79, 45, 33, 5	מספר אי-זוגי קטן מ - 100
(5 + r.Next(45)) * 2	46, 24, 98	מספר דו-ספרתי זוגי
(50 + r.Next(450)) * 2	232, 456, 666	מספר תלת-ספרתי זוגי
r.Next()	18085423,9675543	מספר חיובי כלשהו

ביחידה זו למדנו :

Abs ✓

Pow ✓

Sqrt ✓

Round ✓

Floor ✓

Min ✓

Max ✓

• מהי הספרייה Math

• פעולות בספרייה Math

• קבועים במחלקה Math

• הטיפוס Random להגרלת ערכים

