

.NET

- יש ליצור אפליקציית קונסול NET. עם שלוש פונקציות הבאות:
 - : FileCreateAndFillRandom פונקציה.
- void FileCreateAndFillRandom(string filePath) חתימת הפונקציה:
 - פעולת הפונקציה:
- (1 Gibibyte = 2³⁰ bytes) 1GiB בגודל **filePath** פונקציה יוצרת קובץ חדש בנתיב
 - random bytes ממלא את הקובץ בבתים אקראיים
 - 2. פונקציה CalculateSumSingleThreaded:
 - void CalculateSumSingleTreaded(string filePath) חתימת הפונקציה:
 - פעולת הפונקציה:
- (array of bytes) לתוך מערך הבתים **filePath** פונקציה קוראת את הקוב γ שהועבר לה כפרמטר
 - מאתחלת Stopwatch כדי למדוד זמן
 - מחשבת סכום של כל הבתים בקובץ
 - שימו לב: פונקציה מחשבת סכום של הבתים (sum of bytes) כערך מספרי
 - השתמשו בטיפוס **long** ולא ב
 - בסוף הריצה פונקציה מדפיסה לקונסול הודעה הבאה:

Console.WriteLine(\$"Milliseconds Elapsed: {sw.ElapsedMilliseconds}");
Console.WriteLine(\$"Result: {sum}");



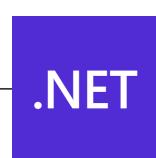


:CalculateSumMultiThreaded פונקציה 3

- void CalculateSumMultiThreaded(object? obj) חתימת הפונקציה:
 - פעולת הפונקציה:
 - הפונקציה מיועדת לרוץ על תהליכון נפרד
 - Parameter פונקציה מקבלת אובייקט \mathbf{obj} וממירה אותו לטיפוס

```
internal class Parameter
{
    public string FilePath { get; set; }
    public int Start { get; set; }
    public int End { get; set; }
    public long PartialSum { get; set; }
}
```

- כל תהליכון שיריץ את הפונקציה יחשב סכום של הבתים בקובץ החל מבית מספר Start
 - ועד הבית מספר End בקובץ FilePath
 - את התוצאה של החישוב יש לשמור במשתנה PartialSum
 - פונקציות **NET**. שאתם צריכים:
 - File.OpenRead(String) Method (System.IO) | Microsoft Learn •
- FileStream.Seek(Int64, SeekOrigin) Method (System.IO) | Microsoft Learn
 - <u>Stream.ReadExactly Method (System.IO) | Microsoft Learn</u> •





- תריצו את שתי הפונקציות הראשונות
- תרשמו תוצאות\הדפסות של הפונקציות כהערה בקוד
 - **•** דוגמה:

```
const string FilePath = @"D:\file.bin";
FileCreateAndFillRandom(FilePath);
CalculateSumSingleTreaded(FilePath);
//Milliseconds Elapsed: 910
//Result: 136903539491
```

- תריצו פונקציה CalculateSumMultiThreaded באמצעות שלושה תהליכונים •
- תמדדו את הזמן שלקח לשלושה תהליכונים לבצע את הפונקציה ותוסיפו כהערה לקוד
 - המטרה שלכם בעצם לחלק עבודה חישובית לשלושה תהליכונים

code booster 🔐 🖋





```
const int _1GiB = 1 << 30;
Thread t1 = new(CalculateSumMultiThreaded),
    t2 = new(CalculateSumMultiThreaded),
    t3 = new(CalculateSumMultiThreaded);
var parameters = new List<Parameter>
    new() { FilePath = FilePath, Start = 0, End = _1GiB / 3 },
    new() { FilePath = FilePath, Start = _1GiB / 3, End = (_1GiB / 3) * 2 },
    new() { FilePath = FilePath, Start = (_1GiB / 3) * 2, End = _1GiB }
var sw = Stopwatch.StartNew();
t1.Start(parameters[0]); t2.Start(parameters[1]); t3.Start(parameters[2]);
t1.Join(); t2.Join(); t3.Join();
```

```
Milliseconds Elapsed: 591
Result: 136903539491
```