

סיכום החומר שלמדנו עד כה

- אלגוריתם - סידרת הוראות מדוייקת ומפורטת מספיק עבור המשתמש, על מנת לבצע פעולה מסוימת. למשל: אלגוריתם לעשיית קפה נמס
 1. הרתח מים בעזרת קומקום שמספיקים לפחות לכוס אחת
 2. שים כפית של קפה נמס בספל
 3. שפוך מהקומקום מים לתוך הספל עם כפית הקפה.
 4. הוסף מעט חלב.
 5. ערבב היטב.
 6. סיים.
- תכונות האלגוריתם: מדוייק וחד משמעי, מפורט מספיק, סופי, פותר בעיות מאותו סוג עם נתוני התחלה שונים, למשל אלגוריתם לחיבור 3 מספרים צריך לתת תשובה נכונה לכל צירוף של 3 מספרים. הנתונים ההתחלתיים של האלגוריתם ניקראים: קלט.
- אלגוריתם יכול להיות מבוסס על הסבר מילולי (כמו ההוראות לעיל), או על ידי תרשים עם צורות שמייצגות פעולות שונות וחיצים שמראים את כיוון זרימת הפעולות - ניקרא תרשים זרימה.
- מושגים בסיסיים השגורים בפי מי שיודע תכנות מחשבים.
 1. מערכת הפעלה - Operating System או בקיצור OS - זוהי תוכנית שבדרך כלל מותקנת על המחשב בזמן הקנייה שלו והיא מאפשרת לאנשים לתקשר עם המחשב. היא למשל שולטת על המקלדת ועל המסך.
 2. דוגמאות למערכות הפעלה שקיימות היום: Windows, Mac OS, Unix, Linux
 3. תוכנית מחשב זהו בעצם קובץ שניתן 'לבצע'. הביצוע של התוכנית ניקרא: הרצה (Run)
 4. ישנן שפות תכנות רבות, אחת מהן שאנחנו נילמד היא: **סי שרפ (ניכתב: C#)**. בשפת התכנות ישנן פקודות שאת התחביר שלהן (Syntax) נילמד בהמשך וגם נילמד מה כל פקודה מבצעת כמובן ומתי להשתמש בה.
 - בצורה הפשוטה ביותר ניתן לומר שתוכנית מחשב היא תירגום אלגוריתם כלשהו לשפה שהמחשב מבין, כלומר מי שיודע לתאר את הפעולות הפשוטות של כל צעד באלגוריתם, יוכל לכתוב תוכנית שתיישם את האלגוריתם.

- הרעיון של פיתרון בעיה בעזרת מחשב יכול להיות מורכב ומעניין. אחרי שהגענו לאלגוריתם הנכון, הכתיבה של התוכנית זהו תהליך מכני לא קשה, אם אנחנו בקיאים בשפת התכנות.
- בלימוד תכנות קורה שלא מבינים דברים מסוימים ורק מניחים שהם עובדים כפי שאנו מצפים. בשלבים מאוחרים יותר מתבהרת התמונה ואז חוזרים ומבינים את החלקים שלא הבנו בהתחלה. זה אומר שהלימוד אינו תמיד מ-אלף עד תו, כי אחרת זה יהיה יבש מידי ומשעמם ולכן עושים קיצורי דרך. אם לומדים בצורה טובה, בסופו של דבר הכל הופך להיות מובן וברור.
- למדנו על המבנה הסכמטי הכללי של המחשב שיש בו יחידת עיבוד מרכזית - cpu, יש לו זיכרון ויש לו ציוד היקפי נילווח (כמו מקלדת, מסך, עכבר ודיסק קשיח). ואמרנו שתוכניות רצות תמיד בזיכרון והמעבד זה מי שמבצע אותן.
- דוגמה לכך ראינו בתוכנית הראשונה שהשתמשה בפקודה: Console.WriteLine - לא הבנו מה זה Console וכעת אנו יודעים שזוהי מחלקה של השפה C# שיש בה פעולות שבהן אפשר להשתמש (גם ReadLine נימצאת בתוכה)
- **משתנים - הקצאת שטח בזיכרון שיש לו שם והוא יכול להכיל ערכים.** למשל מספרים שלמים, מספרים עם נקודה עשרונית, תוים או מחרוזות (אוסף של תוים). להגדיר משתנה, קודם מציינים את סוגו ואחר כך את שמו.
- שמות משתנים מורכבים מאותיות ומספרים בדרך כלל, לדוגמה: num1. אסור שיהיה בתוכם רווחים, ומומלץ שהיו משמעותיים. משתנים מייצגים את אבני הבניין שאתם נבנה את התוכנית.
- דוגמה לתוכנית שכבר ראינו:

```
using System;
```

```
namespace ourprograms
```

```
{
```

```
    class MainClass
```

```
    {
```

```
        public static void Main(string[] args)
```

```
        {
```

```
            Console.WriteLine("Hello World!");
```

```
        }
```

```
    }
```

```
}
```

הסבר של מה שראינו:

כללי, כל המילים שמופיעות בתכלת, הן מילים ששייכות לשפה C# (ניקראות 'מילים שמורות').

המשפט הראשון: `using System;` אומר למחשב שאנו מעוניינים להשתמש בתוכנות שקיימות כחלק מהשפה של C# - הפקודה מסתיימת בתו 'נקודה פסיק'

המשפט: `namespace ourprograms` בעצם אומר שכל מה שיבוא לאחוריו יהיה שייך לאוסף של תוכניות שניקרא: `ourprograms`. שימו לב שלאחרי שורה זו נפתחות סוגריים מסולסלות - זה אומר שניפתח קטע או בלוק של תכנות, שיסתיים כאשר יופיע סוגר מסולסל, כרגע נכתוב תוכנית אחת בלבד.

לאחר מכן ישנה שורה: `class Mainclass` - השם `Mainclass` זה שם שאפשר לשנות (משהו שאנחנו מגדירים). המילה `class` היא מילה 'שמורה' של השפה C#

השם `Mainclass` הוא בעצם שם התוכנית שלנו. שימו לב שוב לסוגריים המסולסלות שמראות היכן מתחילה התוכנית והיכן היא מסתיימת. סביבת העבודה מסדרת לנו יפה לכל פותח { סוגר שמתאים מתחתיו } (בצורה מקוננת).

אחרי השורה הבאה: `public static void Main(string[] args)` היא בעצם המקום שבו התוכנית שלנו מתחילה.

למדנו את הפקודה: `Console.ReadLine()` שקוראת מהקלט ועוצרת את התוכנית. כדי לקרוא מספר שלם 'עטפנו' פקודה זו ב `int.Parse`

לדוגמה:

```
int number1 = int.Parse(Console.ReadLine());
```

למדנו גם שאם לא ניתן פקודת הדפסה לפני הקריאה, המשתמש לא יידע מה להקיש, לכן לרוב לפני `ReadLine` יהיה `WriteLine` שמסביר מה התוכנית מבקשת.

אמרנו גם שצריכים לסיים פקודה עם התו נקודה-פסיק (לא נקודה ופסיק - התו ;)

למדנו מהי פקודת "השמה", שבה ערך מצד ימין של סימן השווה ניכנס למה שיש מהצד השמאלי. למשל:

```
int num1;
int num2;
num1 =7;
```

הערך 7 יכנס למשתנה `num1` לאחר פקודה זו

```
num2= num1 + 5;
```

הערך 12 יכנס לתוך `num2` לאחר פקודה זו

אמרנו שהפקודות מתבצעות אחת אחרי השניה מלמעלה למטה ולכן אם הגדרנו משתנה בשורה השניה ובראשונה ניסינו להשתמש בו, הוא לא היה מוכר.

כתבנו תוכניות שביקשו קלט, ונתנו פלט, כמו להסב טמפרטורה מפרנהייט לצלזיוס. למדנו גם לחשב חישובים שדומים לפעולות מתמטיקה וכתבנו תוכנית שפותרת משוואה ריבועית וגם תוכנית שמחשבת ממוצע של שני מספרים.

למדנו שאם רוצים לכתוב הערה בתוכנית יש שתי דרכים. אחת זה לכתוב שני קוים נטויים מימין לשמאל // ומימנם זו שורה שהמחשב מתעלם ממנה ומיועדת לנו. או להקיף הערה של יותר משורה אחת בתוים: /* ו */ לדוגמה:

```
/*  comments
   .
   .
   .
   */
```

הדבר האחרון שעליו הרחבנו, הייתה מחלקה שמובנית בשפה, בשם Math, שבה ישנן פעולות מתמטיות. השתמשנו בה למצוא שורש של מספר ומקסימום בין שני מספרים. השמוש בה הוא מהצורה:

Math.Action()

כאשר Action זו פעולה (כמו Sqrt או Max) ובתוך הסוגריים ישנם משתנים או מספרים קבועים. הפעולה מחזירה ערך כלשהו. במקרה של Sqrt את השורש הריבועי ובמקרה של Max את הערך הגדול בין שניים.

שיעורי בית

מהספר: יסודות תורת המחשב (עצמים תחילה) עמוד 65 שאלות: 11, 12