**דגשים להעברת הפרויקט**

* תכנון לפני כתיבת קוד ותכנון מודולרי של קוד

**שיטות לתכנון פרוצדורלי**

גישת top-down

פירוק בעיית העל לתתי גורמים - במקרה שלנו מתואר התפריט הנדרש עם רשימת פעולות שניתן לבצע על נתוני הקלט.

עבור כל סעיף יש לפרק את המרכיבים שלו לתתי מרכיבים וכך ממשיכים עד שמגיעים לפעולות פשוטות.

ה - main תקרא לפונקציה המציגה את התפריט.   
הפונקציה המציגה את התפריט, תקרא בתורה לפונקציות השונות המבצעות את הפונקציונאליות הנדרשת עבור כל סעיף, וגם תטפל בקלט עבור הבחירה ובדיקתו.

וכך באופן היררכי נפרט את כל הפונקציות הנדרשות.

באופן זה ניתן לזהות פעולות שחוזרות על עצמן, או פעולות שכדאי לבודד (פונקציות עזר )

גישת bottom-up

בגישה זו נגדיר את הפעולות הפשוטות, ונחבר אותן כמו אבני לגו - עד שנגיע לתוצר הסופי

גישה היברידית (ומקובלת) שילוב של שתי הגישות.

רפרנסים:

<https://www.tomercode.com/2019/09/top-down-programming.html>

[https://adacomputerscience.org/concepts/procprog\_paradigm](https://adacomputerscience.org/concepts/procprog_paradigm#:~:text=The%20procedural%20programming%20paradigm%20encourages,task%20that%20can%20be%20solved)

<https://essensys.ro/2024/03/12/bottom-up-design-explained/>

בתכנון יש להדגיש עבור התלמידים את DRY, Single Responsibility Principle, [Low coupling and high cohesion](https://www.geeksforgeeks.org/software-engineering-coupling-and-cohesion/)

* עבודה עם IDE
  + דיבגר, צפיה בערכי משתנים בזמן ריצה ו call stack

<https://learn.microsoft.com/en-us/visualstudio/debugger/debugger-feature-tour?view=vs-2022>

* + העברת ערכים חיצוניים ל ARGS בהגדרות הפרויקט

<https://dotnettutorials.net/lesson/command-line-arguments-in-csharp/>

* סגנון כתיבה, תעוד, הערות ומוסכמות קוד
* סיוע ופתרון בעיות במהלך הפיתוח
* דגשים לעבודה למול גיטהאב (קומיט לכל פונקציה לדוגמה)
* קצת על בדיקות ידניות ואיך לבדוק כל פונקציה בנפרד

פרויקט אישי

ללא צאט

ללא מתודות מקלות

ניתן להשתמש ב SORT BUILT IN - ניתן לאתגר את המובילים למימוש עצמי

**יעדים**

שעה - לסיום רשימת פונקציות כוללת (בדף ולא בקוד)

* יש לכלול חתימות מלאות (טיפוס מוחזר ופרמטרים)
* מומלץ לכתוב תאור תמציתי מילולי על הביצוע

שעה וחצי - יש לבנות שלד ראשוני ללא מימוש:

* תפריט עם אינדיקציית פלט לכל סעיף ( place holders לפונקציות שיבואו)

ממשיכים עם מימוש הפונקציות השונות

* מימוש בסיסי ללא בדיקות קלט ורק לראות פונקצינאליות עובדת
* מימוש סופי שכולל ולידציות, פיצול לתתי פונקציות, קוד מתועד, זיהוי מקומות שבהם הקוד חוזר בין פונקציות שונות.

בדיקה של כל פונקציה בנפרד - מומלץ ב main ליצור קוד בדיקה זמני לטובת כל פונקציה (או ניתן גם לפתוח פרויקט נפרד למטרה זו)