בס"ד

מיני פרויקט במערכות חלונות

תרגיל מספר 1

מטרת התרגיל

היכרות עם מחלקות ומבנים בשפת #C
Methods ,Fields ,Properties
תכנון נכון של מחלקות
עבודה עם מחלקות קיימות
עבודה עם מערכים
עבודה עם מספרים רנדומליים ותאריכים

הנחיות לביצוע התרגיל והגשתו

- ✔ העבודה תתבצע בזוגות בלבד
- ✔ חובה להשתמש בכלי לניהול גרסאות GIT ובאתר github.org
- ✔ התרגיל יתבצע באותו המאגר ובאותו ה-Solution כמו תרגיל המבוא (תרגיל 0)
- ✔ יש להגיש במודל קישור על פי ההנחיות בקובץ "הגשת מטלות בקורס מיני פרויקט במערכות חלונות 153007"!
- ✔ נא להקפיד על פורמט זה על מנת למנוע מצב של אי קבלת ציון על תרגיל מסוים

המטלה

הפרויקט השנה עוסק בניהול מערכת מידע של שירות משלוחים המתבצעים באמצעות רחפנים, המשלוחים מתבצעים בין הלקוחות השונים של השירות.
למשל משלוחים מחנויות ועסקים שונים לצרכנים השונים.
בתרגילים המקדימים נתחיל לבנות את הישויות בהן נשתמש במהלך הפרויקט.

בתרגיל זה נתחיל להגדיר את הישויות שנשתמש בפרויקט.
בנוסף נבנה מערכת זמנית שתספק תפריט ותקלוט מידע מהמשתמש ולאתחל את המערכת בנתונים, שימו לב שניתן ליצור את התפריט בממשק קונסול (טקסטואלי).

המשלוחים מתבצעים בין המשתמשים (לקוחות) של החברה שנותנת את שירות המשלוחים.
החברה מחזיקה תחנות-בסיס עבור תחזוקת וטעינת הרחפנים וכן מידע על לקוחות וחבילות שנשלחו באמצעות השירות.

החברה המפעילה את שירות המשלוחים רוצה לעקוב אחר ביצוע המשלוחים וכן לנהל את המעקב על תחזוקת הרחפנים. בהמשך הפרויקט יהיה מעקב רחפנים וחבילות.

בתרגיל זה עליכם להגדיר מערכים בגודל קבוע מראש על מנת לייצג את הרשימות של הישויות השונות ולאכלס אותם לפי קלט מהמשתמש.
התוכנית הראשית בשלב זה תציג תפריט למשתמש עם האפשרויות הבאות (על הסטודנטים לנתח את הישויות ולהסיק מסקנות, יש לשים לב שאין לכלול לוגיקה בתוכנית הראשית וגם לא במתודות גישה\הוספה\עדכון נתונים):

1. אפשרויות הוספה:
   - הוספת תחנת-בסיס לרשימת התחנות
   - הוספת רחפן לרשימת הרחפנים הקיימים

- קליטת לקוח חדש לרשימת הלקוחות
- קליטת חבילה למשלוח.

2. אפשרויות עדכון:
   - שיוך חבילה לרחפן
   - איסוף חבילה ע"י רחפן
   - אספקת חבילה ל-לקוח
   - שליחת רחפן לטעינה בתחנת-בסיס
     - ע"י שינוי מצב הרחפן והוספת רשומה (מופע) של ישות טעינת סוללת רחפן
     - התחנה נבחרת ע"י המשתמש בתוכנית הראשית (כדאי שהמשתמש יבחר מתוך רשימת תחנות-בסיס פנויות - ראה אפשרויות הצגת רשימות)
   - שחרור רחפן מטעינה בתחנת-בסיס

3. אפשרויות תצוגה (כולן ע"פ מספר מזהה מתאים):
   - תצוגת תחנת-בסיס
   - תצוגת רחפן
   - תצוגת לקוח
   - תצוגת חבילה

4. אפשרויות הצגת הרשימות
   - הצגת רשימת תחנות-בסיס
   - הצגת רשימת הרחפנים
   - הצגת רשימת הלקוחות
   - הצגת רשימת החבילות
   - הצגת רשימת חבילות שעוד לא שויכו לרחפן
   - הצגת תחנות-בסיס עם עמדות טעינה פנויות

5. יציאה

ישויות הנתונים:

לפניכם דוגמה (לא מחייבת מבחינת השמות) להגדרת הישויות:

- תחנת-בסיס
  - מספר מזהה ייחודי
  - שם התחנה
  - מספר עמדות הטענה *(פנויות)*
  - קו אורך (longitude)
  - קו רוחב (latitude)

- רחפן
  - מספר מזהה ייחודי
  - מודל הרחפן
  - קטגוריית משקל (קל, בינים, כבד)
  - מצב סוללה (רמת טעינה)
  - מצב הרחפן (פנוי, תחזוקה, משלוח)

- לקוח
  - מספר מזהה ייחודי
  - שם הלקוח
  - מספר טלפון
  - קו אורך (longitude)
  - קו רוחב (latitude)

- חבילה
  - מספר מזהה ייחודי
  - מזהה לקוח שולח
  - מזהה לקוח מקבל
  - קטגוריית משקל (קל, בינוני, כבד)
  - עדיפות (רגיל, מהיר, חירום)
  - מזהה רחפן מבצע (0 אם לא הוקצה)

- זמן יצירת חבילה למשלוח
- זמן שיוך החבילה לרחפן
- זמן איסוף חבילה מהשולח
- זמן הגעת החבילה למקבל

נ.ב. הוספת זמנים מציינת התקדמות החבילה יצירה => שיוך => איסוף => אספקה

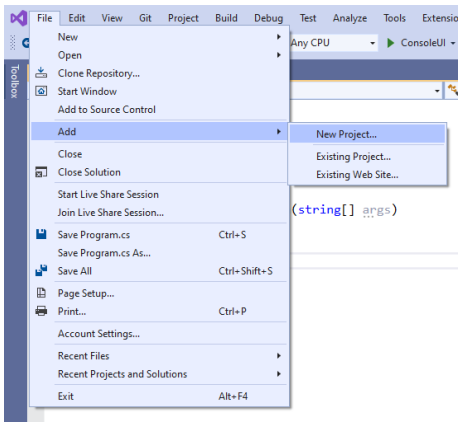
- טעינת סוללת רחפן
  - מזהה תחנת-בסיס
  - מזהה רחפן

<u>שימו לב</u>: תפקיד ישויות הנתונים הינו שמירת מידע ולא לוגיקת מערכת. כל התכונות נדרשות בהמשך לפרויקט וללוגיקה שלו, גם אם כרגע לא לגמרי ברור לכם למה אנחנו צריכים חלק מהן.

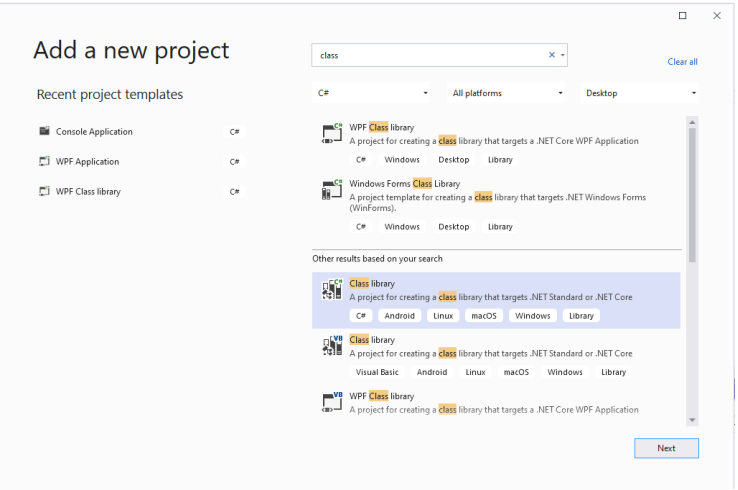
הנחיות ליצירת פרויקטים ותלויות בין פרויקטים בתרגיל

התרגיל יתבצע באותו הפתרון (Solution) שנפתח בתרגיל 0 בשני פרויקטים חדשים - ConsoleUI ו-DAL. הפרויקט ConsoleUI יכיל הפניה (יהיה תלוי ב-) לפרויקט DAL, כדלקמן.

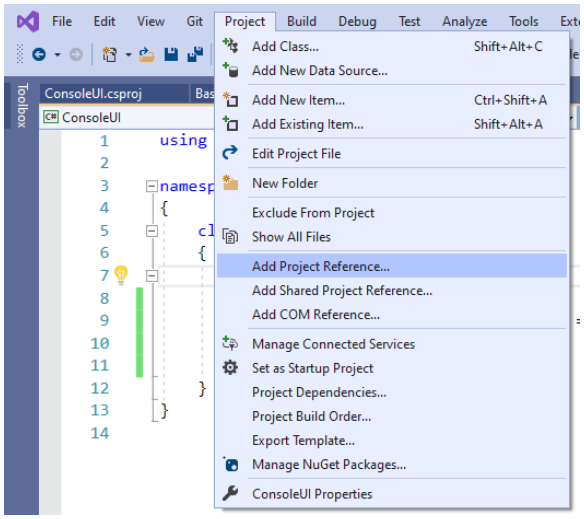
1. הוספת פרויקט חדש לSolution:

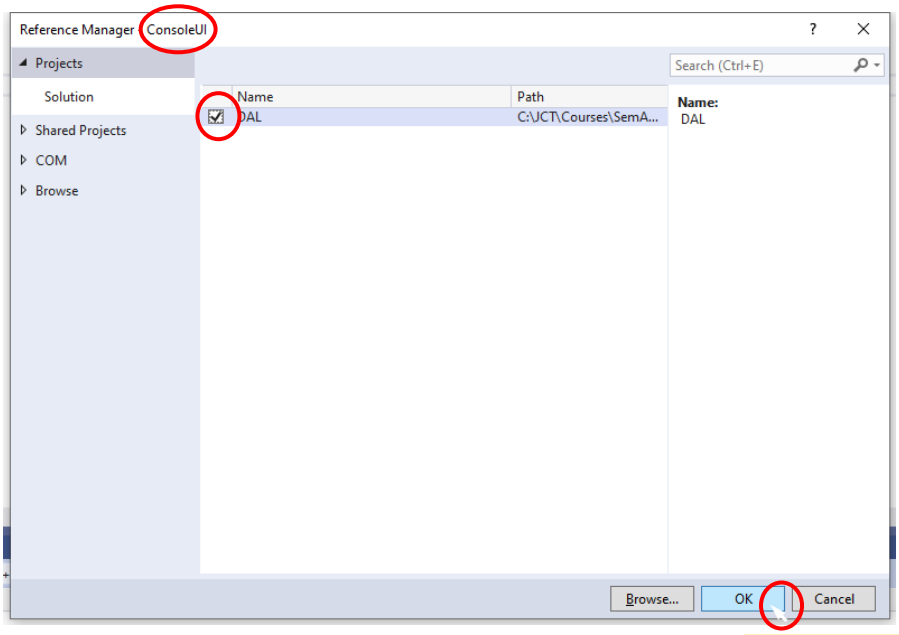


2. נוסיף Project מסוג Console בשם ConsoleUI עבור התוכנית הראשית של התרגיל
3. נוסיף Project מסוג Class Library בשם DAL עבור ישויות הנתונים

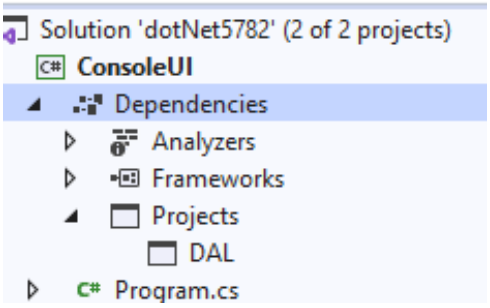


4. כאשר אנחנו נמצאים בפרויקט ConsoleUI נוסיף לפרויקט של התכנית הראשית Project Reference





5. נבחר את הפרויקט אליו נרצה להתייחס



6. אפשר לראות בחלונית Solution Explorer שנוסף לרשימת התלויות
7. עכשיו אפשר לעבוד עם ישויות שהוגדרו בתוך מרחבי השמות בפרויקט השני

```
using System;

namespace ConsoleUI
{
    class Program
    {
        static void Main(string[] args)
        {
            IDAL.DO.BaseStation baseStation = new IDAL.DO.BaseStation();
            Console.WriteLine(baseStation);
        }
    }
}
```

הנחיות לביצוע התרגיל:

- יש להשתמש במבנים (struct) על מנת לממש את ישויות הנתונים.
- התוכנית הראשית תוגדר בפרויקט בשם ConsoleUI, התוכנית תקבל את הנתונים בעזרת הפונקציות המתוארות בהמשך וידפיסו את התוצאות - התוכנית הראשית לא תכלול שום לוגיקה למעט קלט, בקשות נתונים ופלט.
- הפלט בתוכנית הראשית יהיה ע"י פלט ישיר של האובייקטים של הישויות.
- ישויות הנתונים הנ"ל:
  - בפרויקט חדש ששמו DAL מסוג בתוך מרחב שמות (namespace) מקונן IDAL.DO
    נ.ב. דוגמא של הגדרת מרחבי שמות מקוננים:

```
namespace IDAL
{
        namespace DO
        {
                public struct ...
        }
}
```

  - יוגדרו כ-PDS, זאת אומרת Passive Data Structure כדלקמן:
    - כל ישות תוגדר במודול (קובץ #C) נפרד
    - הגדרה כמבנה (struct) עם הרשאה public
    - הנתונים יוגדרו כתכונות בהגדרה מהירה ועם הרשאה public, למשל:

```
public string Name { get; set; }
```

    - רק מתודה אחת (לצורכי הדפסה בתוכנית הראשית של קונסול ו-debug) - דריסת ToString עם החזרת מחרוזת בפורמט ברור וקריא
- כל האנומרציות (enum) הנדרשות עבור הישויות יוגדרו במודול אחד נפרד בפרויקט ששמו DAL בתוך מרחב שמות (namespace) מקונן IDAL.DO, כנ"ל.
- בפרויקט DAL במרחב שמות DalObject תוגדר מחלקה DataSource:
  - המחלקה תכיל מערכים סטטיים של ישויות הנתונים הנ"ל בהרשאה internal.
  - המערכים יאותחלו בהגדרתם בגודל כדלקמן:
    - עד 10 רחפנים
    - עד 5 תחנות-בסיס
    - עד 100 לקוחות
    - עד 1000 חבילות
  - המחלקה תכיל מחלקה פנימית (מקוננת) בשם Config בהרשאה internal
    - המבנה יכיל שדות סטטיים עבור מציינים (אינדקסים) של האלמנט הפנוי הראשון בכל אחד מהמערכים בהרשאה internal
    - השדות יאותחלו לערך 0
    - בנוסף יהיה שדה עבור יצירה של מספר מזהה רץ עבור חבילות (ראה בהמשך)
  - המחלקה תכיל מתודה סטטית בשם Initialize שתאתחל באתחול מהיר את מופעי הישויות במערכים עם נתונים וגם עדכון שדות במחלקה Config בהתאם:
    - לפחות שתי תחנות-בסיס
    - לפחות 5 רחפנים במצבים שונים
    - לפחות 10 לקוחות
    - לפחות 10 חבילות במצבים שונים
    - חלק מהנתונים יוגרלו בעזרת Random ע"פ הגיון בריא
    - מספר רץ לחבילה יאותחל לערך שיהיה גדול מכל מספרי החבילות שנוספו לעיל
    נ.ב. כדאי שמצב הנתונים יהיה מסודר ושלם (זאת אומרת, שלא יהיה רחפן במצב תחזוקה באמצע משלוח, למשל)

- כל מתודות הוספת\הבאת\עדכון הנתונים יוגדרו כדלקמן:
  - תוגדר מחלקה DalObject בפרויקט DAL במרחב שמות DalObject
  - המחלקה הנ"ל תוגדר עם הרשאת public

- כל המתודות יוגדרו עם הרשאת public
- כל המתודות יעסקו בהבאת נתונים ולא יכללו שום לוגיקה של בדיקות תקינות וכדומה
- גישה למופעי כל הישויות תהיה בעזרת מפתח - המספר המזהה הייחודי, ולא ע"י מצייני (אינדקסי) האלמנטים של המערכים
- הוספת נתונים תתבצע לפי המציין הראשון הפנוי - ראו בסעיפי הבאים
- הבנאי של המחלקה יפעיל אתחול למחלקת DataSource בעזרת המתודה הסטטית DataSource.Initialize()
- כל המתודות המחזירות רשימות ישויות יבנו מערך חדש מתאים בגודל לפי כמות הישויות המועברות, ויעתיקו את האובייקטים מהמערכים ב-DataSource
- מתודת הוספת חבילה תיצור מספר חבילה אוטומטית על בסיס המספר הרץ כנ"ל, תקדם את המספר הרץ במחלקת Config ותחזיר את המספר הנוצר

<u>הנחיות כלליות</u>:

- יש לפרמט את הקוד בהתאם - הזחות, שורות רווח, ריווח בתוך השורות.
- כל השמות (של המחלקות, השדות, התכונות, המתודות) חייבות להיות באנגלית ובעלי משמעות בהתאם לתפקיד המחלקה, המבנה, התכונה, המתודה.
- על השמות להיות מוגדרים בפורמט CamelCase עבור כל הסוגים, השדות, התכונות והמתודות עם הרשאה public או internal.
- על השמות להיות מוגדרים בפורמט camelCase עבור שדות ומתודות בהרשאה private או protected ועבור משתנים מקומיים.
- חובה לתעד את המתודות בעזרת תיעוד מפורמט (///)
- חובה לתעד את הקוד אם איננו ברור מאליו ממבנה הקוד (למשל במקרה של חישובים מתמטיים שאינם בסיסיים, אלגוריתם שאיננו ברמה של "פשיטא ליה"
- בתוכנית הראשית לצורך המרות עבור הקלט יש להשתמש בפונקציה TryParse
- יש להפריד את התוכנית הראשית לתתי פונקציות

<u>אפשרויות בונוס</u>:

- תצוגה על בסיס סֶקְסָגֵסִימָלִי (**sexagesimal** - על בסיס 60) של ערכי הקואורדינטות, למשל:
  - קו אורך
    -36.123456 => 36° 7′ 24.441″ S
  - קו רוחב
    29.654321 => 29° 39′ 15.555″ E
- אפשרות נוספת בתפריט שקולטת קואורדינטות נקודה כלשהי ומדפיסה מרחק מבסיס או מלקוח כלשהו לנקודה הזו
- תעלו רעיונות ותבקשו מהמרצה שלכם\ן להעביר לרכזי הקורס - נשקול ונאשר (ונפרסם) או לא נאשר:
  - הרעיון צריך לחדש ולא סתם להוסיף מה שדומה למה שכבר הוגדר
  - רעיון צריך להישאר במסגרת החומר שנלמד עד לתרגיל 1 מבחינת תחביר **C#** וכלים של דוט נט

נ.ב. כל בונוס המאושר ע"י רכזי הקורס יוסיף נקודה אחת לציון הסופי של רכיב התרגול בציון הקורס.

נ.ב. בקבוצות שנלמדו אוספים (כולל List) לפני פתיחת התרגיל ניתן להשתמש ב-List במקום המערכים ב-DataSource. במקרה כזה אין צורך להשתמש במצייני המקום הפנוי הראשון במערכים ואפשר לא לכלול את השדות האלה במחלקת Config.

נספחים:

על הסטודנטים ללמוד על המבנה **DateTime** ועל השימוש בו (כולל בנאים רלוונטיים).
קבלת התאריך הנוכחי ע"י המבנה **DATETIME**:
ע"י שימוש במאפיין **Now** של המבנה **DateTime** ניתן לקבל את התאריך הנוכחי

```
DateTime currentDate = DateTime.Now;
```

הגרלת מספר רנדומלי ע"י המחלקה **RANDOM**:

הקצאת משתנה שבאמצעותו נגריל את המספרים הרנדומליים **הערה**: מומלץ לייצר כשדה סטטי ולהשתמש בהמשך לכל הצרכים. וזאת למניעת יצירת אובייקטים חדשים כל הזמן רק עבור כמה מספרים רנדומליים.	**Random r = new Random();**
מגריל מספר שלם חיובי בטווח של 32Int	**r.next();**
מגריל מספר שלם חיובי עד ולא כולל max	**r.next(max);**
מגריל מספר בטווח בין low (כולל) לבין high (לא כולל)	**r.next(low, high);**

בהצלחה רבה!