בס"ד תשפ"ג

מיני פרויקט במערכות חלונות

שלב 1

# מטרות התרגיל

היכרות עם מחלקות ומבנים בשפת #C

Properties, Fields, Methods

תכנון נכון של מחלקות

עבודה עם מחלקות קיימות

מערכים

מספרים רנדומליים

תאריכים ופרקי זמן

הנחיות לביצוע התרגיל והגשתו

* העבודה תתבצע בזוגות בלבד
* חובה להשתמש בכלי לניהול גרסאות GIT ובאתר github.org
* חובה לבצע את השלב באותו המאגר ובאותו ה-Solution כמו שלב 0 (מקדים)
* חובה להגיש במודל קישור על פי הנחיות ההגשה
* חובה להקפיד על פורמט זה על מנת למנוע מצב של אי קבלת ציון על תרגיל מסוים
* חובה לקרוא את התיאור הכללי של הפרויקט לפני תחילת העבודה על השלב

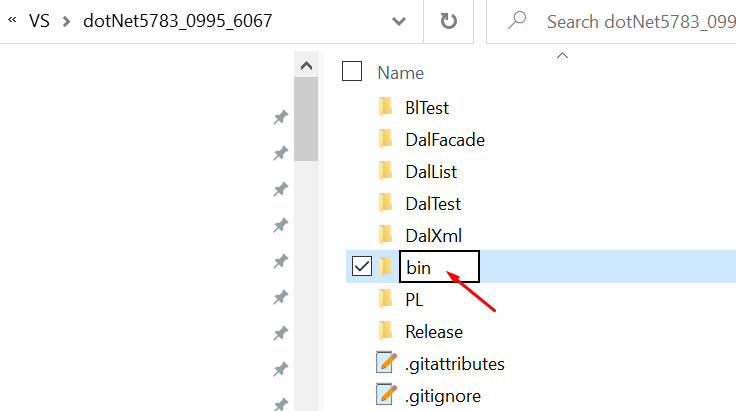
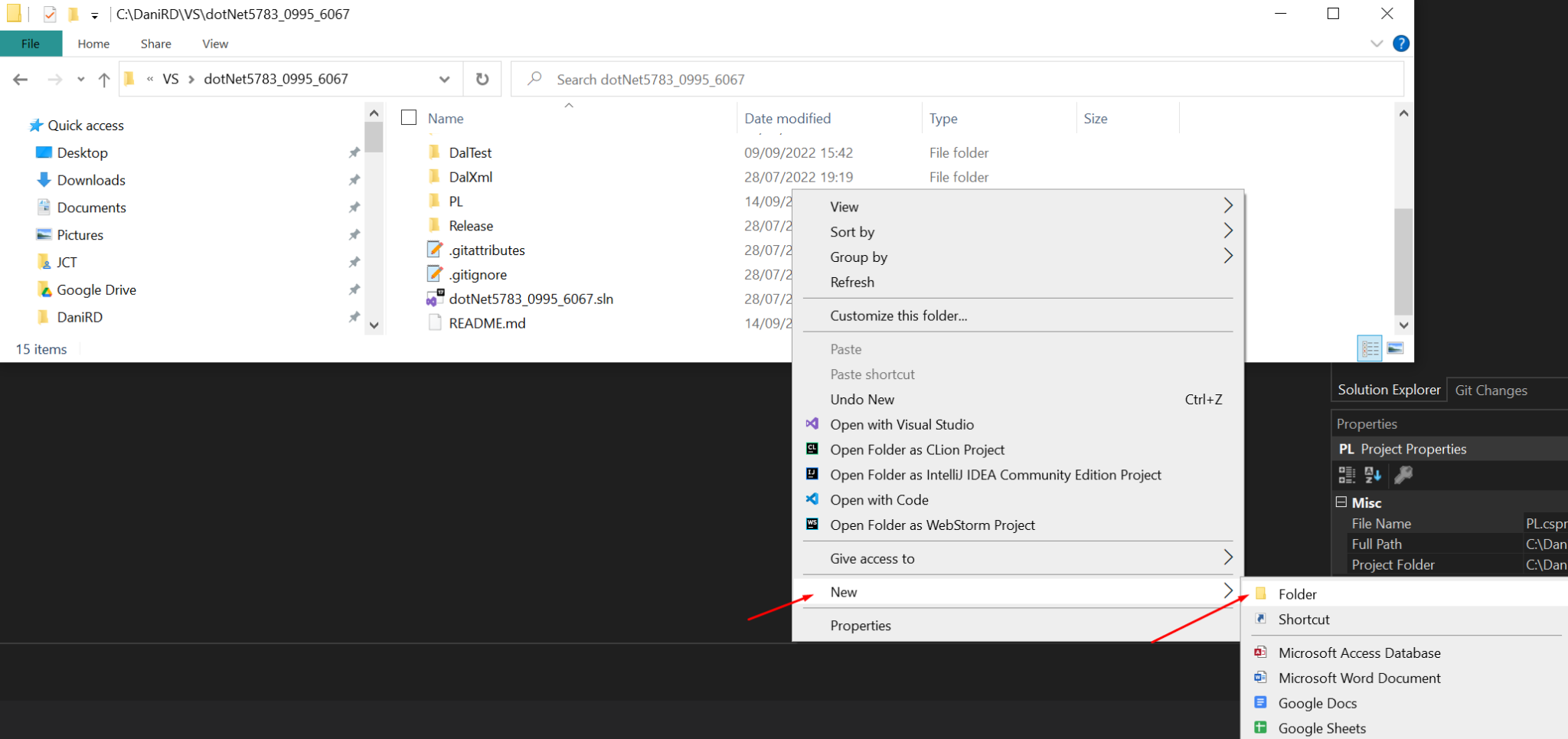
**המטלה**

בשלב הזה נתחיל לבנות את שכבת הנתונים של הפרויקט - נגדיר את **ישויות הנתונים** וכן מחלקות עם הפונקציות הרלוונטיות לעבודה עם הישויות כפי שמתואר בהמשך. לצדן נגדיר מחלקה שתהיה אחראית להחזקת הנתונים במערכים ועל אתחול הראשוני שלהם בתחילת ההרצה.

בנוסף נבנה ממשק משתמש כקונסול טקסטואלי (CLI) שיספק תפריט ויקלוט מידע מהמשתמש על מנת לבצע את הפעולות בעזרת המתודות הנ"ל. תפקיד ממשק המשתמש של השלב הזה הוא להוות **כלי\תוכנית לבדיקה** ידנית (manual test program) של תפקוד שכבת הנתונים הנבנית בשלב הזה.

בתרגיל זה עליכם להגדיר מערכים בגודל קבוע מראש על מנת לייצג את הרשימות של הישויות השונות ולאכלס אותם גם בעת האתחול בצורה אוטומטית (באתחול האוטומטי חובה לדאוג להתאמה מלאה בין הנתונים) וגם לפי קלט מהמשתמש (כאן התאמה של הנתונים הינה באחריות המשתמש המפעיל את תוכנית הבדיקות).

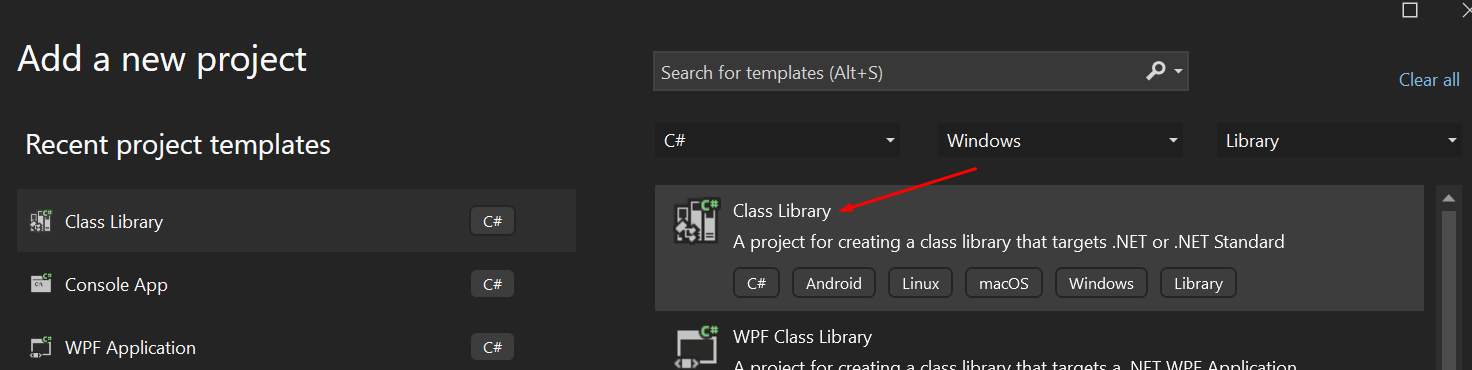
**הכנת התיקיות והפרויקטים לשלב הנוכחי**

בתיקיה הראשית של ה-Solution יש ליצור תת-תיקיה בשם bin (היצירה חייבת להתבצע בסייר קבצים - File Explorer, בשום פנים ואופן **אסור** "ליצור" את התיקיה מתוך תפריט הקשר של ה-Solution) - בתחילה תיווצר תיקיה חדשה בשם "New folder" ויש לעדכן את שמה ל-"bin":  


הוספת תיקייה bin וחלק מההגדרות המתוארות בהמשך נעשות על מנת לרכז את כל קבצי הריצה במרוכז במקום אחד בתיקייה הראשית של Solution ולמנוע חלק מהבעיות ומקושי של מעקב אחר קבצי ריצה בהמשך הדרך.

עכשיו יש להוסיף ל-Solution שלושה פרויקטים (במהלך הוספה יש לוודא שהפרויקטים נפתחים עם Framework בגרסה NET 6.0.).

* הוספת פרויקט DalFacade מסוג Class Library:



* + עשו לחיצה כפולה על שם הפרויקט ב-Solution Explorer (ייפתח לעריכה קובץ ניהול הפרויקט עם סיומת csproj) - יש להוסיף לו את השורות הבאות (המודגשות) בסוף האלמנט PropertyGroup:

*<Project Sdk="Microsoft.NET.Sdk">*

*<PropertyGroup>*

*<TargetFramework>net6.0</TargetFramework>*

*<ImplicitUsings>enable</ImplicitUsings>*

*<Nullable>enable</Nullable>*

**<BaseOutputPath>$(SolutionDir)\bin\</BaseOutputPath>**

**<OutputPath>$(SolutionDir)\bin\</OutputPath>**

**<AppendTargetFrameworkToOutputPath>false</AppendTargetFrameworkToOutputPath>**

**<AppendRuntimeIdentifierToOutputPath>false</AppendRuntimeIdentifierToOutputPath>**

**<UseCommonOutputDirectory>true</UseCommonOutputDirectory>**

**<RootNamespace />**

*</PropertyGroup>*

*</Project>*

* + בתפריט ההקשר של הפרויקט (לחיצה על כפתור ימני כאשר הסמן מעל שם הפרויקט) בחרו מאפיינים (Properties) וודאו שהשדה Default namespace ריק (תחת כותרת Application>General, כתוצאה מהשורה המודגשת האחרונה לעיל), ובשדה Base output path (תחת כותרת Build>Output, כתוצאה מהשורה המודגשת הראשונה לעיל)
  + **נ.ב.** יתר ההוספות שעשינו לעיל לא משתקפות בממשק המאפיינים של פרויקט ולכן עשינו אותם ע"י עריכה ישירה בתוך הקובץ csproj
* הוספת פרויקט DalList מסוג Class Library:
  + עשו לחיצה כפולה על שם הפרויקט ב-Solution Explorer (ייפתח לעריכה קובץ ניהול הפרויקט עם סיומת csproj) - יש להוסיף לו את השורות הבאות (המודגשות) בסוף האלמנט PropertyGroup:

*<Project Sdk="Microsoft.NET.Sdk">*

*<PropertyGroup>*

*<TargetFramework>net6.0</TargetFramework>*

*<ImplicitUsings>enable</ImplicitUsings>*

*<Nullable>enable</Nullable>*

**<BaseOutputPath>$(SolutionDir)\bin\</BaseOutputPath>**

**<OutputPath>$(SolutionDir)\bin\</OutputPath>**

**<AppendTargetFrameworkToOutputPath>false</AppendTargetFrameworkToOutputPath>**

**<AppendRuntimeIdentifierToOutputPath>false</AppendRuntimeIdentifierToOutputPath>**

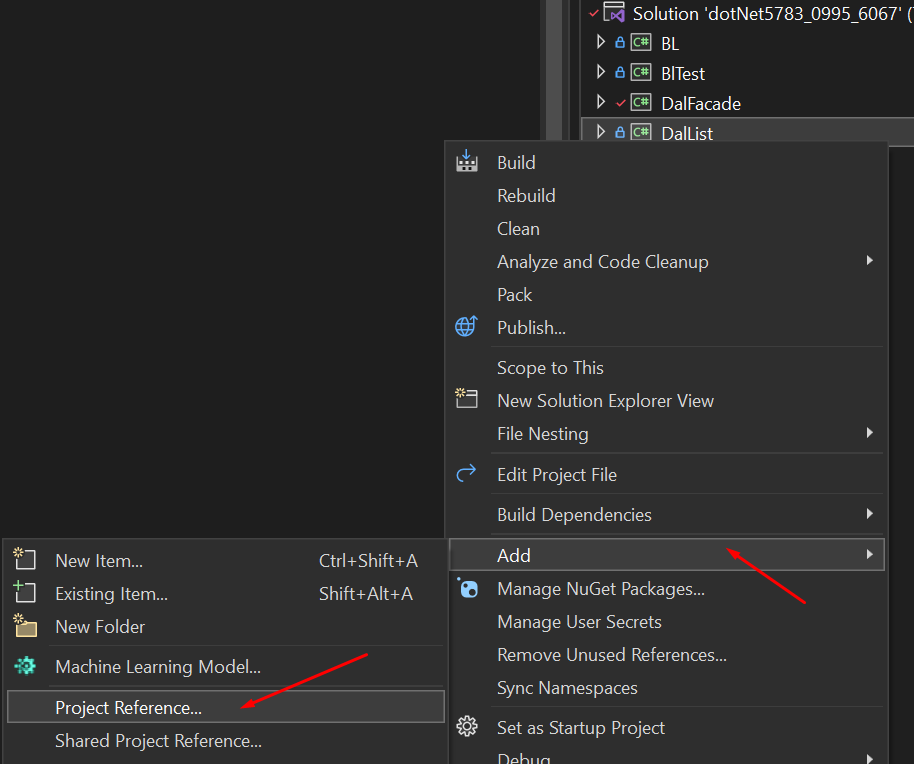
**<UseCommonOutputDirectory>true</UseCommonOutputDirectory>**

**<RootNamespace>Dal</RootNamespace>**

*</PropertyGroup>*

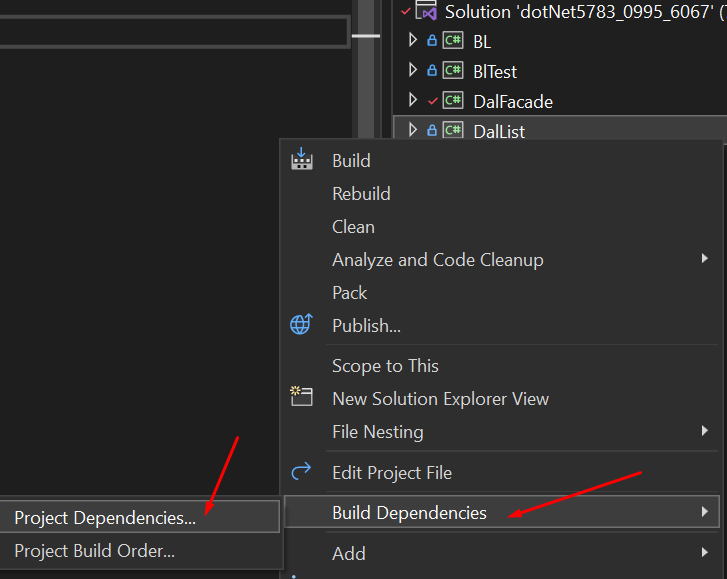
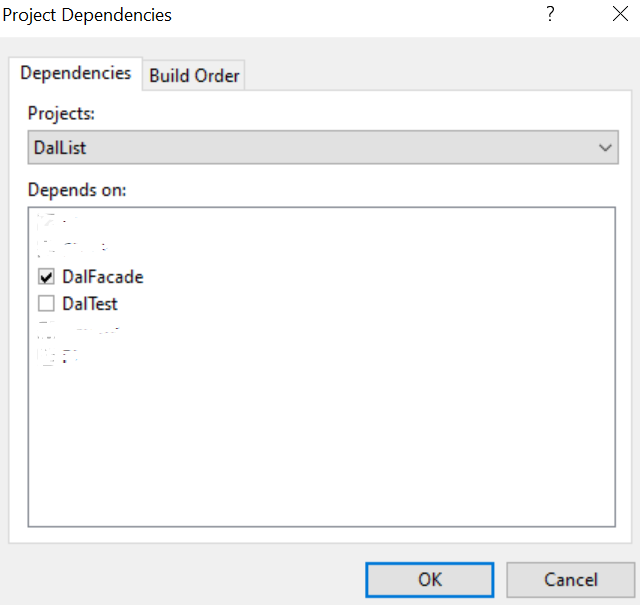
*</Project>*

* + בתפריט ההקשר של הפרויקט בחרו מאפיינים (Properties) ותוודאו שהשדה Default namespace מכיל "Dal" (תחת כותרת Application>General, כתוצאה מהשורה המודגשת האחרונה לעיל), ובשדה Base output path (תחת כותרת Build>Output, כתוצאה מהשורה המודגשת הראשונה לעיל)
  + **נ.ב.** יתר ההוספות שעשינו לעיל לא משתקפות בממשק המאפיינים של פרויקט ולכן עשינו אותם ע"י עריכה ישירה בתוך הקובץ csproj
  + בתפריט ההקשר של הפרויקט בחרו הוספת הפניות פרויקט (...Project Reference):

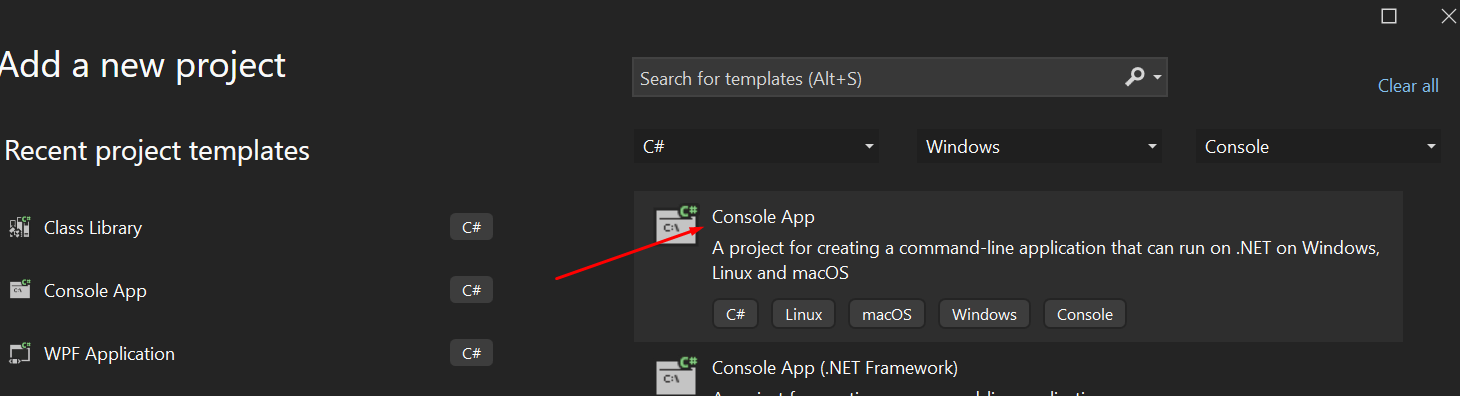


ובו סמנו את הפרויקט DalFacade:  (לחצו OK לאחר מכן)

* + בתפריט ההקשר של הפרויקט בחרו …Build Dependencies>Project Dependencies וודאו שהסימון דומה לסימון ההפניות כנ"ל (סמנו אם הסימון חסר, הצילום נעשה בשלב מתקדם יותר של הפרויקט ולכן חלקים מסוימים הוסרו ע"י צביעה על מנת לא להפריע לכם כרגע):

* הוספת פרוייקט DalTest מסוג Console App:



* + עשו לחיצה כפולה על שם הפרויקט ב-Solution Explorer (ייפתח לעריכה קובץ ניהול הפרויקט עם סיומת csproj) - יש להוסיף לו את השורות הבאות (המודגשות) בסוף האלמנט PropertyGroup:

*<Project Sdk="Microsoft.NET.Sdk">*

*<PropertyGroup>*

*<OutputType>Exe</OutputType>*

*<TargetFramework>net6.0</TargetFramework>*

*<ImplicitUsings>enable</ImplicitUsings>*

*<Nullable>enable</Nullable>*

**<BaseOutputPath>$(SolutionDir)\bin\</BaseOutputPath>**

**<OutputPath>$(SolutionDir)\bin\</OutputPath>**

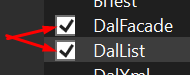
**<AppendTargetFrameworkToOutputPath>false</AppendTargetFrameworkToOutputPath>**

**<AppendRuntimeIdentifierToOutputPath>false</AppendRuntimeIdentifierToOutputPath>**

**<UseCommonOutputDirectory>true</UseCommonOutputDirectory>**

*</PropertyGroup>*

*</Project>*

* + בתפריט ההקשר של הפרויקט בחרו מאפיינים (Properties) ותוודאו שהשדה Default namespace מכיל "Dal" (תחת כותרת Application>General, כתוצאה מהשורה המודגשת האחרונה לעיל), ובשדה Base output path (תחת כותרת Build>Output, כתוצאה מהשורה המודגשת הראשונה לעיל)
  + **נ.ב.** יתר ההוספות שעשינו לעיל לא משתקפות בממשק המאפיינים של פרויקט ולכן עשינו אותם ע"י עריכה ישירה בתוך הקובץ csproj
  + בתפריט ההקשר של הפרויקט בחרו הוספת הפניות פרויקט (...Project Reference), ובו סמנו את הפרויקטים DalFacade ו-DalList: 
  + בתפריט ההקשר של הפרויקט בחרו …Build Dependencies>Project Dependencies וודאו שהסימון דומה לסימון ההפניות כנ"ל (מסומנים שני הפרויקטים DalFacade ו-DalList).

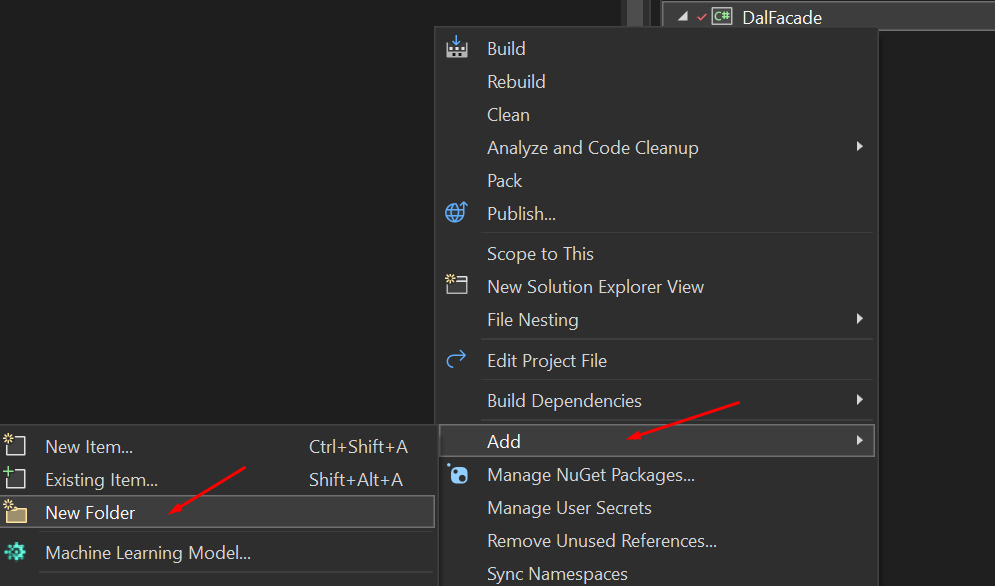
**הנחיות ביצוע השלב**

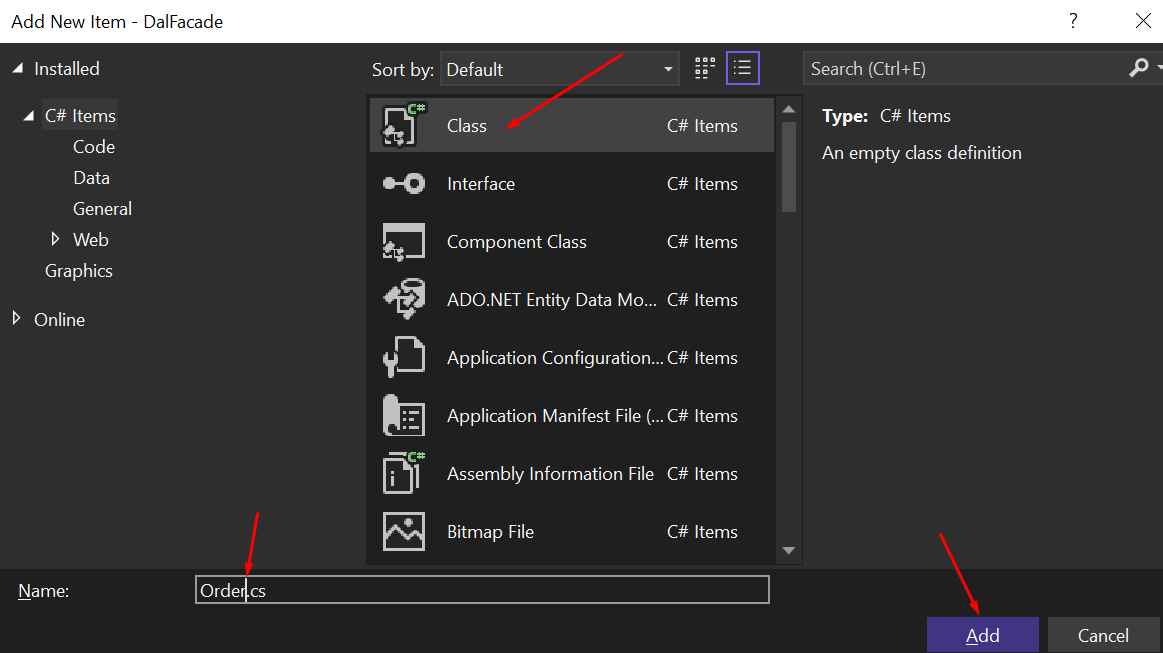
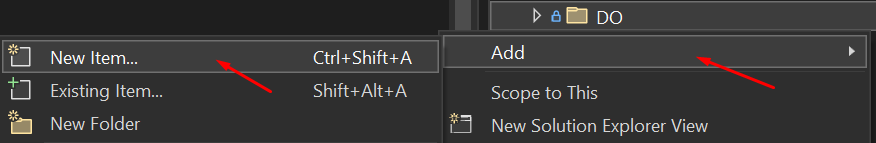
**הנחיות כלליות**:

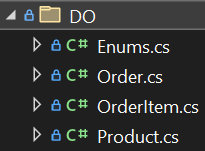
* חובה לפרמט את הקוד בהתאם - הזחות, שורות רווח, ריווח בתוך השורות
* כל השמות (של המחלקות, השדות, התכונות, המתודות) חייבות להיות באנגלית ובעלי משמעות בהתאם לתפקיד המחלקה, המבנה, התכונה, המתודה
* שמות כל הסוגים, כמו כן השדות, התכונות, והמתודות עם הרשאה public יש להגדיר בפורמט PascalCase
* שמות כל השדות והתכונות בהרשאה private או internal יש להגדיר בפורמט camelCase\_
* שמות כל המתודות בהרשאה private או internal, הפרמטרים, והמשתנים המקומיים יש להגדיר בפורמט camelCase
* חובה לתעד את כל הסוגים, המתודות, והתכונות 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"

**ישויות הנתונים:**

צרו בפרויקט DalFacade תת-תיקיה בשם DO:

 =>

* בתיקיה הזו צרו מחלקה עבור כל ישות ובנוסף קובץ קוד עבור כל האנומרציות (enum):  
  



הנחיות כלליות לגבי כל קבצי ישויות נתונים (קבצים ב-DO):

* יש להשתמש במבנים (struct) על מנת להגדיר את ישויות הנתונים
* כל ישויות הנתונים יהיו עם הרשאה public
* בכל הקבצים של DO תמחקו את כל ה-using (אם ישנם, ראו שהם מופיעים בצבע "דהוי" ובעצם כבר קיימים במרומז)
* בכל הקבצים של DO הוסיפו סימן ";" בסוף השורה: "**;**namespace DO", שימו לב שהצומדיים של הבלוק **{...}** של מרחב השמות נמחקים אוטומטית ע"י ויז'ואל סטודיו, וכל התוכן של הבלוק מוזח שמאלה בהתאם. זהו החידוש בגרסה 10 של #C.

כל האנומרציות (enum) הנדרשות עבור הישויות יוגדרו בקובץ אחד Enums.cs כנ"ל

כל ישות נתונים תוגדרו כ-PDS, זאת אומרת Passive Data Structure כדלקמן:

* כל ישות תוגדר במודול (קובץ #C) נפרד
* כל ישות תהיה עם הרשאה public ולכולן יש לשנות את המחלקה למבנה: מ-class ל-struct

namespace DO;

/// <summary>

/// Structure for ...

/// </summary>

public struct ...

* כל התכונות יוגדרו כתכונות אוטומטיות (**{;**get**;** set**}**) עבור כל הנתונים של הישות, כפי שמוגדר במסמך תיאור הפרויקט

/// <summary>

/// Unique ID of ...

/// </summary>

public int ID { get; set; }

* הוסיפו **דריסה** של מתודה ToString שתחזיר מחרוזת המתארת אובייקט, למשל:

public override string ToString() => $@"

Product ID={ID}: {Name},

category - {Category}

Price: {Price}

Amount in stock: {InStock}

";

* אסור לכתוב כל מימוש של תכונות או כל מתודה במבנים של ישויות נתונים למעט המתודה ToString הנ"ל
* הוסיפו הערות למבנה ולכל תכונה של מבנה בעזרת הערת תיעוד (תוסיפו שורה ריקה לפני הגדרת מבנה או תכונה ותקלידו בשורה החדשה שלושה סלאשים /// ותוסיפו את תיעוד המבנה\התכונה), למשל:

**מימוש שכבת נתונים:**

שכבת הנתונים תמומש בשלב הזה ע"י מערכים של מבני ישויות נתונים. כל הקוד יהיה בפרויקט DalList במרחב שמות Dal.

צרו את המחלקות המממשות את שכבת הנתונים:

* עבור כל ישות נתונים צרו מחלקת גישה לנתונים בשם ####Dal עם הרשאה public, כאשר #### הוא שם ישות הנתונים בהתאם
* צרו מחלקה **סטטית** DataSource עם הרשאה internal (מה ההבדל?)
* בכל הקבצים הנ"ל תמחקו את כל ה-using (אם ישנם, ראו שהם מופיעים בצבע "דהוי" ובעצם כבר קיימים במרומז)
* בכל הקבצים הנ"ל הוסיפו סימן ";" בסוף השורה: "**;**namespace Dal"

במחלקה DataSource:

* כל האיברים המחלקה יהיו סטטיים מכיוון שהמחלקה סטטית
* הוסיפו למחלקה שדה סטטי readonly עבור יצירת מספרים רנדומליים שיאותחל בהגדרה
* המחלקה תכיל מערכים של ישויות הנתונים הנ"ל בהרשאה internal
  + המערכים יאותחלו בהגדרתם בגודל כפי שמוגדר בתיאור הכללי של פרויקט
  + עבור כל ישות, צרו מתודה פרטית שתוסיף אובייקטים למערך של הישות
  + צרו מתודה פרטית בשם s\_Initialize, המתודה תזומן מבנאי ברירת המחדל (הסטטי) של המחלקה
  + המתודה s\_Initialize תזמן את מתודת הוספת אובייקטים למערכי הישויות לפי סדר נכון בהתאם לתלויות בין הישויות
  + כל הנתונים של הישויות חייבים להיות הגיוניות ומתואמות כפי שמופיע בתיאור הכללי של הפרויקט
    - נתוני התאריכים שבישויות:
      * כל התאריכים החסרים (מטיפוס DateTime) בנתוני הישויות יאותחלו ל-DateTime.MinValue
      * כל התאריכים שיש ביניהם סדר - יש להשתמש ב-TimeSpan עם פרק זמן מוגרל רנדומלית (ע"פ **היגיון בריא**) שיוסף לתאריך "הקודם" לפי משמעות התאריכים בישות הרלוונטית
    - מספרים מזהים של ישויות בכל אובייקט חדש:
      * מספר מזהה רץ אוטומטי יתקבל מתכונה מתאימה במחלקה הפנימית Config
      * יתר מספרים מזהים יוגרלו רנדומלית בהתאם למה שכתוב בתיאור הכללי של הפרויקט
    - יצירת חלק מהנתונים תתבצע בצורה רנדומלית ע"פ **הגיון בריא,** יש לשמור על שלמות הנתונים
  + המחלקה תכיל מחלקה סטטית פנימית (מקוננת) בשם Config בהרשאה internal
    - הוסיפו למחלקה שדות סטטיים בהרשאה internal עבור מציינים (אינדקסים) של האלמנט הפנוי הראשון בכל אחד ממערכי הישויות - כמובן השדות יאותחלו ל-0
    - הוסיפו למחלקה שדות סטטיים פרטיים עבור מספר מזהה אחרון עבור הישויות **שיש להם מזהה רץ אוטומטי** (השדות יאותחלו למספר מזהה הקטן ביותר לפי דרישות כל ישות)
    - עבור כל שדה של מספר מזהה אחרון, הוסיפו תכונה עם get בלבד שתקדם את השדה אוטומטית (ב-1) כך שכל פעם יתקבל מספר שגדול מהקודם ב-1

בכל מחלקות גישה לנתוני הישויות

* הוסף מתודות בסיסיות של גישה לנתונים ע"פ שיטת CRUD:
  + כל המתודות יהיו בהרשאה public
  + מתודת הוספת אובייקט שתקבל אובייקט של ישות ותחזיר את המספר המזהה של האובייקט שנוסף
    - אם לא מדובר בישות עם מספר מזהה רץ אוטומטי, יש לבדוק שהאובייקט עם המספר המזהה הזה עוד לא קיים
    - אובייקט חדש יתווסף במקום הפנוי הראשון במערך ע"פ השדה המתאים במחלקה הפנימית Config שב-DataSource
  + מתודת בקשה\קריא של אובייקט בודד שתקבל **מספר מזהה** של הישות (***שימו לב - לא מדובר במציין\אינדקס במערך!***) ותחזיר את האובייקט המתאים
  + מתודת בקשה\קריאה של רשימת כל האובייקטים של הישות (ללא פרמטרים)
  + מתודת מחיקת אובייקט של ישות שתקבל מספר מזהה של הישות
  + מתודת עדכון אובייקט שתקבל אובייקט חדש
    - המתודה תדרוס את האובייקט הישן ע"י האובייקט החדש באותו מקום במערך
    - בתחילת העדכון יש לוודא שהאובייקט קיים - לפי מספר מזהה
* הוסף מתודות נוספות ע"פ הדרישות בתיאור הכללי של הפרויקט
  + כל המתודות יהיו בהרשאות public
* אסור לכלול במתודות שום לוגיקה (למשל לוגיקת בדיקות תקינות וכדומה)
  + הלוגיקה המותרת היחידה היא של חיפוש האובייקט או האובייקטים בהתאם לתפקיד מתודה ובדיקת תקינות תוצאות החיפוש
  + במקרה של תקלה כלשהי (למשל אובייקט לא נמצא, או אובייקט כבר קיים - במקרה של הוספה) יש לזרוק חריגה ע"י ***("טקסט קצר של תיאור הבעיה")throw new Exception***
* כל המתודות המחזירות רשימות ישויות **יבנו מערך חדש** מתאים **בגודל לפי כמות הישויות המועברות**, ויעתיקו למערך המוחזר את האובייקטים מהמערכים ב-DataSource

**תוכנית ראשית:**

התוכנית הראשית של השלב תציג תפריט למשתמש עם האפשרויות הבאות (על הסטודנטים לנתח את הישויות ולהסיק אלו פרטים נדרשים בכל אפשרות, יש לשים לב ש**אין לכלול שום לוגיקה** בתוכנית הראשית או במתודות של שכבת הנתונים).

בתפריט הראשית של התוכנית יוצגו כאפשרויות בדיקות עבור כל ישות נתונים. לאחר בחירת ישות מסוימת תיפתח תת-תפריט לביצוע אחת הפעולות של הישות - תהיה אפשרות נפרדת עבור כל מתודה במחלקת גישה של היישות. למשל כדלקמן:

0. יציאה

1. בדיקת *שם-ישות-1*:
   1. אפשרות הוספת אובייקט לרשימה של ישות
   2. אפשרות תצוגת אובייקט ע"פ מזהה
   3. אפשרות תצוגת הרשימה של ישות
   4. אפשרות עדכון נתוני אובייקט
   5. אפשרות מחיקת אובייקט מרשימת של ישות
2. בדיקת *שם-ישות-2*:

…..

כמובן שאם לישות יש מתודות בנוסף ל-CRUD - יש להוסיף אפשרויות עבור הבדיקות שלהן - **חובה להתייחס לכל המתודות**.

* התוכנית הראשית תוגדר בפרויקט DalTest
* במחלקה של התוכנית הראשית יוגדרו שדות פרטיים עבור כל אחד ממחלקות גישה לישויות
  + השדות יאותחלו בהגדרה עם אובייקט של מחלקה מתאימה
* המתודה Main תעבוד בלולאה שתדפיס בקונסול את התפריט הראשית, תקלוט את הבחירה ותפעיל את תת-תפריט המתאים
* הוסף מתודה נפרדת עבור תת-תפריט של כל ישות ובה:
  + תדפיס את רשימת הבדיקות עבור הישות
  + תקלוט את בחירת המשתמש
  + תקלוט את הנתונים המתאימים עבור המתודה הנבדקת
  + תזמן את המתודה הנבדקת
  + אם המתודה הנבדקת מחזירה ערך - תדפיס את הערך המוחזר
  + אם המתודה הנבדקת זרקה חריגה - תתפוס את החריגה ותדפיס את ההודעה שבחריגה
* כדי להעביר את הקלט למתודות הוספה יש ליצור את האובייקט המתאים להעבירו בארגומנט
* בבדיקת מתודות עדכון יש לקלוט מספר מזהה, להדפיס את האובייקט שהתקבל ולבקש לקלוט ערכים חדשים
  + במקרה של קלט ריק - הנתון לא יעודכן ויישאר כפי שהיה
* בהדפסות:
  + אובייקטים של ישויות נתונים יודפסו בעזרת מתודת ToString ישירות
    - **אסור לזמן את המתודה ToString בצורה מפורשת!**
  + מערך יודפס בעזרת לולאת foreach ובתוכה הדפסת אובייקטים כנ"ל
* **התכנית לא תכלול שום לוגיקה למעט קלט, בקשות נתונים (זימון מתודות) ופלט**
* **לבונוס**: לצורך המרות של מספרים ותאריכים מהקלט - להשתמש בפונקציה TryParse

**בהצלחה רבה!**