**תיאור כללי:**

עליכם לתכנן ולממש מערכת פשוטה לניהול סניף בנק. המערכת תכלול ניהול של לקוחות, חשבונות בנק מסוגים שונים, וסניפים.

**🎯 מטרות הלמידה:**

* תרגול ירושה בין מחלקות.
* שימוש ב- **Overload**, **Virtual**, **Override**, **Protected**
* שימוש בממשקים ומחלקות מופשטות (abstract)
* הבנה והמחשה של עקרונות OOP דרך תרגול פרקטי.
* תכנון UML וביצוע מימוש לפי הדיאגרמה.

**דרישות:**

**1. תכנון: UML**

לפני כל מימוש בקוד, תכנן דיאגרמת UML של המחלקות המרכזיות במערכת:

* קשרי ירושה וממשק
* תכונות (שדות)
* פעולות (מתודות)
* סוגי הגישה (public, protected, private)

**2. מחלקות בסיסיות שצריכות להופיע:**

המערכת צריכה לכלול לפחות את המחלקות הבאות:

**א. BankBranch (סניף בנק)**

* branchNumber (מספר סניף)
* address (כתובת)
* - accounts מערך של 20 CheckingAccount
* פעולה המוסיפה חשבון למערך חשבונות – AddAccount (BankAccount account)
* פעולה המדפיסה החשבונות בסניף - PrintAccounts()

**ב. Customer (לקוח)**

* id (ת"ז)
* name (שם)
* phoneNumber (טלפון)
* פעולה - PrintDetails()

**ג. BankAccount (מחלקת בסיס לחשבון בנק)**

* accountNumber (מספר חשבון)
* balance (יתרה)
* customerOwner (בעל החשבון מסוג Customer)
* פעולת הפקדה – Deposit (double amount)
* פעולת משיכה - Withdraw(double amount)
* פעולה PrintBalance() – (תהיה virtual)

**ד. CheckingAccount יורש מ- BankAccount**

**CheckingAccount** (בעברית: **חשבון עו"ש**) הוא סוג של חשבון בנק שבו הלקוח יכול לבצע פעולות יומיומיות כמו הפקדות, משיכות, והעברות, לרוב ללא מגבלות כמו בחשבון חיסכון.

* תכונה נוספת - overdraftLimit
* פעולה Withdraw(double amount) – מאפשרת משיכה גם אם עוברים למינוס עד גובה מסוים
* פעולה PrintBalance() – override עם הדפסת יתרת מינוס מותרת

**3. שימוש בטכניקות OOP:**

* השתמש ב-**Protected** עבור תכונות שיש לגשת אליהן מתוך מחלקות יורשות.
* בצע שימוש ב-**Overload** עבור מתודות כמו (Deposit למשל הפקדה עם תאריך)
* בצע שימוש ב-**Virtual/Override** עבור PrintBalance().
* כל החשבונות במערך accounts ב- BankBranch יהיו מסוג CheckingAccount.
* הוסף ממשק IBankBranch שתכלול את כל המתודות של BankBranch

**4. כללים נוספים:**

* מותר ורצוי להוסיף שדות או פעולות נוספות לפי ראות עיניכם (בתנאי שנשמר עקרון הירושה וההיגיון האובייקטי).
* יש לכתוב **תוכנית ראשית (Main)** שמדגימה שימוש במחלקות השונות.
* השתמשו ב- Get ו- Set הרגיל שלמדתם, ולא באלה המובנים של VS
* תשתמשו רק בכלים שלמדתם. אין להשתמש בכלים אחרים שלא נכללו בחומר הלימוד.
* אין להשתמש בבינה מלאכותית, הקוד ייבדק ובמידה ונעשה שימוש בבינה מלאכותית יורדו נקודות

**5. אופן הגשה:**

1. **UML**: קובץ PDF או תמונה של דיאגרמת המחלקות.
2. **קוד C# מלא** לפי הדיאגרמה שתכננת. שם הפרויקט **AccountManagement**
3. כל מחלקה/ממשק תיכלל בקובץ נפרד בפרויקט
4. **העלה את הפרויקט ל- GitHub**
5. בקלאס רום, העלה את קובץ ה- UML והקישור למאגר (Repository) של הפרויקט ב- Github.

**6. מועד אחרון להגשה:**

**15/5/2025** – לא יינתנו הארכות מלבד מילואים (עם אישור)

**7. ניקוד**

1. משקל הבוחן הוא 30 נקודות מהציון הסופי
2. יינתנו נקודות בונוס (עד 5 נקודות) על השקעה ומאמץ מעבר לדרישות.