**תרגול 11 – OOP**

* נלמד עקרונות לתכנות
  + כדאי לשאוף אליהם אבל לא תמיד יהיה אפשר לממש אותם
* נדבר על 4 דברים עיקריים:
  + מודולריות
  + עיקרון פתיחה סגירה
  + עיקרון ההחלפה של ליסקוב
  + תבנית עיצוב ממנטו

**Modularity Principle:**

* חלוקת התוכנית לכמה מודולים
* העיקרון אומר שמערכת צריכה להיות מורכבת מרכיבים שונים, שכל אחד מהם עומד בפני עצמו והם מחוברים אחד לשני בצורה לא הדוקה
  + אם יש שני רכיבים קשורים מאוד אז הם כנראה היו צריכים להיות רכיב אחד
* הרכיבים צריכים להיות מסוגלים לתקשר ביניהם אבל לא יכולים להיות יותר מידי תלויים אחד בשני
* מודול צריך להיות משהו שניתן להעביר למקומות אחרים
* יתרונות:
  + קל לטיפול – דיבוג, הרחבה, לעשות טסטים
  + הופך בעיות גדולות לבעיות קטנות יותר – בדר"כ יותר קל להבנה ולדיבוג
  + מאפשר חלוקה של הקוד לבנייה של כמה אנשים במקביל
* בשביל לייצר מודולריות יש את Meyer’s Modularity Criteria – מהי מודולריות ומה צריך לעשות כדי לקבל קוד מודולרי
  + Modular Decomposability - עיקרון הפריקות – שואפים לפרק בעיה לתתי בעיות. בחלוקה כל מודול יפתור בעיה קטנה ומוגדרת וניתן לחבר את כולם כך שיפתרו את הבעיה הגדולה
    - החיבור בין המודולים צריך להיות יחסית מינימלי
  + Modular composability – הרכבה – להשתמש ביחידות קטנות ומורכבות ולחבר אותם ליחידה הגדולה. חיבור המרכיבים הוא בצורה קלה והגיונית
    - חוסך כפל קוד
  + Modular Continuity – שינוי כל קוד במודולים הוא אינדיבידואלי מבלי להשפיע על כל המערכת
    - שינוי ההנחיות יגרום לשינוי ספציפי במודולים
  + Modular Understandability - מובנות – צריכים להיות מסוגלים להסביר את התפקיד של כל מודול.

**Open-Close Principle:**

* רכיבי תוכנה שונים צריכים להיות פתוחים להרחבה אבל סגורים לשינויים
  + כלומר אנחנו צריכים להיות פתוחים להוספת פונקציונליות או הרחבה כלשהי אבל מבלי לשנות את הקוד והמחלקה – רק להוסיף דברים שנבנים על אותו הדבר.

\*\*\* במצגת תרגול – עיקרון ההחלפה של ליסקוב ותבנית עיצוב ממנטו