# **תשובות**

### **עקרון הכימוס (Encapsulation):**

עקרון הכימוס או במילים אחרות Encapsulation זה עקרון אשר מתאר שימוש באובייקטים בידיעה כיצד להשתמש בהם אך הדרך שהם עובדים מאחורי הקלעים מוסתר לנו. בנוסף, לעקרון זה יש מספר יתרונות:

* גמישות וניהול קוד – המתכנתים לא צריכים לדעת את מימוש המחלקה, הפעולות שלה, וכל הדברים הקטנים הסובבים לה אלא רק את מה שהם צריכים באותו רגע ורק איך להשתמש. ברגע שיש שינוי אין צורך להחליף את כל המחלקה אלא רק לעדכן את מימוש הפעולה אשר תלויה/קשורה לשינוי.
* קריא יותר – ניתן לקשר דברים דומים ביחד תחת מחלקה אחת וככה אפשר להימנע מיישומים מסובכים מדי וקוד "זרוק".
* אבטחה טובה יותר – זה יתרון חשוב משום שהוא מביא את כל הרעיון של אבטחה בתכנות מונחה עצמים. ניתן להגביל ואף לתת גישה לתכונות ולפעולות במחלקה למחלקות אחרות.

### **מחלקה לעומת מחלקה אבסטרקטית (Class VS Abstract Class):**

מחלקה או Class זה בעצם האבן בניין של כל הנושא של תכנות מונחה עצמים. המחלקה עצמה מהווה מן בית/קופסא אשר בתוכה נמצאים כל התכונות והפעולות. בנוסף, המחלקה משמשת כתבנית לייצור האובייקטים.

התכונות זה אוסף של משתנים אשר מכונים כ-Attributes והפעולות מכונות כ-Methods. אפשר להסתכל על זה שהתכונות אחראיות על מה היא המחלקה והפעולות על מה אפשר לעשות עם המחלקה.

ברגע שיוצרים אובייקט ניתן לגשת (אם יש הרשאות) לפעולות ולתכונות (נהוג שלא תהיה גישה לתכונות – עקרון הכימוס). בנוסף, למחלקה רגילה ניתן לגשת לפעולות/תכונות סטטיות מבלי ליצור מופע של המחלקה.

עכשיו, ההבדל בין מחלקה רגילה למחלקה אבסטרקטית היא שמחלקה רגילה אפשר לעשות ממנה אובייקט ומחלקה אבסטרקטית זה פשוט יחזיר שגיאה. הסיבה לכך היא שמחלקה אבסטרקטית לא נועדה כדי שייצרו ממנה מופעים או שייגשו לפעולות/תכונות שלה אלא שמחלקות אחרות ישתמשו בה (בפעולות והתכונות) כאבן בניין בעזרת ירושה.

דבר אחרון זה שבמחלקה אבסטרקטית ניתן לממש פעולות אבסטרקטיות בלי גוף וככה לדרוש מהיורשים לדרוס את הפעולה.

### **מחלקה לעומת מחלקה סטטית (Class VS Static Class):**

כפי שהסברתי בשאלה הקודמת על מזה מחלקה ומחלקה אבסטרקטית אפשר לציין שמחלקה סטטית די דומה למחלקה אבסטרקטית משום שאי אפשר ליצור מהן מופעים. אך, ההבדל בין שניהם זה שבמחלקה סטטית אפשר לגשת לתכונות/פעולות של המחלקה.

אם נסתכל על הפרטים הקטנים, ניתן ליצור מופעים ממחלקה רגילה אך אי אפשר ליצור מופעים ממחלקה סטטית. מחלקה סטטית שימושית לפעולות כלליות כמו חישובים למיניהם. הפעולות והתכונות במחלקה סטטית יהיו גם הם סטטיים.

### **Internal Access Modifier:**

הרעיון פה מדבר על הרשאות שזה חלק מנושא עקרון הכימוס (Encapsulation), וישנן כמה סוגי השראות לדוגמה: public, private, protected, and internal.

Public אומר שניתן לגשת מכל מקום (מחלקות אחרות ופרויקטים אחרים), private שניתן לגשת אך ורק מהמחלקה שמשתמשים בהרשאה ו-protected אומר שהמחלקה שמשתמשת בהשראה יכולה לגשת וכל המחלקות שיורשות מאותה מחלקה יכולות לגשת גם כן.

Internal בעצם אומר שרק קוד באותו Assembly או במילים אחרות באותו הפרויקט יכול לגשת למידע הזה. פרויקט לדוגמה זה "Console App". קל יותר להבין את הרעיון מאחורי ההרשאה בעזרת דוגמה:

ניצור שני פרויקטים – A וB. בA יש לנו מחלקה public שאנחנו רוצים להשתמש בB. בתוך B ניתן לגשת למחלקה בA ברגע שנקרא לפרויקט A בעזרת Using בחלק העליון של פרויקט B ואז לממש את המחלקה שאנחנו רוצים וזה יעבוד.

לעומת זאת אם המחלקה בA תהיה internal זה אומר שלא נצליח לגשת למחלקה בפרויקט B גם לאחרי שהשתמשנו בusing, כי internal בעצם אומר שניתן לגשת למחלקה הזאת אך ורק בתוך פרויקט A (Assembly A) ואנחנו קראנו לה בתוך פרויקט B (Assembly B).