**עקרונות וטיפים ב-OOP**

1. מחלקות ועצמים
2. constuctors
3. INSTANCE
4. GET,SET
5. DATA MEMBERS
6. ACCESSORS
7. TOSTRING
8. METHODS
9. OBJECT CLASS
10. Object Array
11. Encapsulation
12. INHERITANCE
13. פולימורפיזם
14. אוספים
15. INTERFACE
16. DELEGATES, EVENTS
17. OPERATOR OVERLOADING
18. Attributes
19. GENRICS
20. STREAMS and FILEs
21. Serialization
22. Linq
23. Abstract class
24. Static class
25. Const
26. Readonly
27. Static member
28. Static constructor
29. Exceptions
30. Boxing and unboxing

נושאים

עקרונות עיקריים של OOP

1. אנקפסוליישן- האפשרות לחשוף או לחשוף מידע פנימי של אובייקט כלפי חוץ.
2. הינהריטנס(הורשה) – האפשרות שאובייקט יירש מידע מאובייקט אחר ,כך שיהיו להם תכונות משותפות וללא צורך בכתיבת קוד נוסף זהה.
3. פולימורפיזם- כאשר אובייקט יורש מאובייקט אחר , קיימת לו האפשרות לממש תכונה שאותה הוא יורש בצורה המתאימה לו ע"י דריסת הפונקציה המקורית.

ההבדלים בין מחלקה אבסטרקטית וממשק

1. בתוך מחלקה אבסטרקטית ניתן לממש פונקציות וממשק לא
2. ניתן לרשת ממספר אינסופי של ממשקים ורק ממחלקה אבסטרקטית אחת( או מחלקה אחת)
3. בתוך ממשק כל הפונקציות הן PUBLIC בברירת מחדל ולא ניתן לשנות את זה , לעומת מחלקה אבסטרקטית שניתן להגדיר פונקציות כלא PUBLIC
4. במחלקה אבסטרקטית ניתן להגדיר שדות , מה שלא ניתן לעשות בממשק

מהו HASH TABLE ומה היתרונות שלו

HASH TABLE הוא טבלה מילונית כשלכל ערך יש מפתח ייחודי . המפתח הייחודי נהפך ע"י Hash function לאינדקס של תא מסויים. כאשר רוצים להכניס/להוציא/לחפש נתון מסויים הפעלת ה-HASH FUNCTION על המפתח תאפשר את מציאת האינדקס של התא וביצוע הפעולה ביעילות

מה זה ג'נריקס

האפשרות ליצור מחלקה או פונקציה עם טיפוס לא מוכר ולתת לו שם רק תוך כדי יצירת האובייקט או שימוש בפונקצייה .

היתרון הוא מניעת מימוש חוזר עבור סוגי טיפוסים שונים

אקספשין

דוגמא עם 3 מקרים



3 המקרים דומים בזה שהאקספשין הוא יהיה אקספשין כללי

מקרה 1 ו2 הם בדיוק אותו הדבר ! הם יזרקו את אותו אקספשין שישמור את המקור שלו ואת הסטאק טרייס.

במקרה 3 האקספשיין לא שומר את המקור שלו ולא את הסטאק טרייס – האקספשין יזרק כאילו הוא קרה בשורה הזאת .



בדוגמא הזאת יש שגיאת DB בתוך הפונקציה SAVE של האובייקט.

האקספשין שתופסים ושזורקים הוא לא אותו אחד . במקרה הזה (להבנתי) ישמר סטאק טרייס חדש ועדיין נשמור את המידע על האקספשין המקורי

לכל אפליקציה טוב שיהיו מספר קטן של application exceptions ששייכים לה. בצורה כזאת המשתמשים באפליקציה יכולים להבין לאיזה אקספשינס אפשר לצפות מהאפליקציה.

האפליקציה לא אמורה לשלוח אקספשין שקשור ללוגיקה של משתנים (לדוגמה) ולכן הוא עטוף באקספשין של המערכת .   
עדיין ה- business exceptions לא יודעים בדרך כלל מידע מפורט על מקור הבעיה ולכן יש אקספשין המקונן ,שדרכו נשמר מידע על מקור הבעייה

דוגמא – שאלת מוסך מעבודה של HIT קורס של גיא רונן