1 - המטרה של CIL היא להיות שפת מהדר בשביל שפות .NET, כלומר, אחרי תהליך הקומפילציה שפות .NET מתורגמות לCIL שיותר קרובה לשפת מכונה. בזמן ריצה שפת הCIL מתורגמת לזמן מכונה. המנגנון הזה נועד לשפר ביצועים - המחשב מתרגם מ CIL ל Assembly הרבה יותר מהר מאשר היה עושה זאת מ C# ל Assembly. תהליך זה מאוד דומה למה שקורה עם Just in time ו Java.

2 - אנחנו נשתמש ב ILDASM כדי לעשות דיקומפילציה לקוד מקומפל. למשל אם אנחנו מקבלים אפליקציה מוכנה שאנחנו רוצים להשתמש בה באפליקציה שלנו, אנחנו יכולים להשתמש ב ILDASM כדי לראות את הקוד ולהבין איך היא עובדת מאחורי הקלעים על מנת לאפשר אינטגרציה טובה יותר.

אנחנו נשתמש ILASM כדי לקמפל קוד בשפת CIL, למשל אם השתמשנו ב ILDASM כדי לעשות דיקומפילציה לקוד של אפליקציה מסוימת ושינינו משהו בקוד, אנחנו נשתמש ב ILASM כדי לקמפל את הקוד חזרה.

3 - שימוש ב ILDASM עוזר לנו בתכנות מונחה עצמים כיוון שבעזרתו אנחנו יכולים לעשות דיקומפילציה לקוד ולראות איך הקוד עובד מאחורי הקלעים. למשל אנחנו יכולים לראות שכשאנחנו קוראים לא מייצרים בנאי אנחנו מקבלים בנאי ריק חינם.

4 - Guest, User, Administrator
5 - הCMD הוא הקונסול של ווינדוס (הטרמינל של לינוקס), אנחנו נשתמש בו כדי להריץ אפליקציות קונסול של C # (אם בכלל :)) או אפליקציות אחרות שאין להן ממשק UI, אם וכשנשתמש בו נשתמש בהרשאת Admin כדי לקבל את הגישה לכל מה שנרצה לעשות (ההרשאה הכי גבוהה שאין לה הגבלות גישה).

6 - שפת IL משתמשת בפקודת newobj כדי ליצור מופע חדש.
7 - מופיע S ליד המתודה PrintGame כיוון שזאת מתודה סטטית.
8 - אנחנו נקרא לWriteLine בCall
9 - אנחנו נקרא לTellMeHowManyPlayers בCallvirt
10 - ההבדל בין בcall לcallvirt הוא שcallvirt ימצא את המיקום של הקוד הרלוונטי רק בזמן ריצה, כלומר נחפש את המתודה המתאימה במחלקת האם ולאחר מכן במחלקות היורשות במידת הצורך כדי שנוכל לממש פולימורפיזם, בזמן שcall י"יודע" את המיקום המדויק של הקוד ואין שום מצב שהמיקום ישתנה בזמן ריצה או יהיה null.

בזמן שcallvirt יזרוק NullReferenceException במידה וthis שהוא מקבל כארגומנט ראשון כשאנחנו קוראים למתודה של אובייקט הוא null, call ניגש ישר לbytecode ומנסה לממש את מה שהוא מצא.