Part02

what's the difference between compiled and interpreted languages and in this way what about Csharp?

**Compiled Languages**: Code is translated into machine code (binary) before running , Generally faster as machine code runs directly on the CPU ,

Errors are detected during compilation (before execution), like C, C++, Rust , Distributes binary executable files.

**Interpreted Languages**: Code is executed line-by-line by an interpreter , Slower because code is interpreted at runtime , Errors are detected during runtime,

Like Python, JavaScript, Ruby , Distributes source code or intermediate representation.

**Compilation to Intermediate Language (IL):**

* When you write C# code, it is compiled into an intermediate language (IL) by the C# compiler (csc.exe).
* This IL code is platform-independent and saved in an assembly file (e.g., .exe or .dll).

**Just-In-Time (JIT) Compilation:**

* At runtime, the **Common Language Runtime (CLR)** converts the IL code into machine code specific to the operating system and processor.
* This process is called **Just-In-Time (JIT) compilation**.

**Benefits of the Hybrid Model:**

* Combines the performance of compiled languages with the flexibility of interpreted languages.
* Allows cross-platform execution by compiling IL to machine code at runtime.

Compare between implicit, explicit, Convert and parse casting?

**Implicit Casting**:

* **What it is**: Automatic conversion performed by the compiler from a smaller data type to a larger one.
* **Example**: Converting int to double.
* **Why it's safe**: There's no risk of data loss because the destination type can hold all values of the source type.

**Explicit Casting**:

* **What it is**: Manual conversion specified by the programmer when converting from a larger data type to a smaller one or between incompatible types.
* **Why it's risky**: There’s potential for data loss or runtime errors if the values don't fit.

**Convert Class**:

* **What it is**: Provides a safe way to convert between data types using methods like Convert.ToInt32().
* **When to use**: When you need to convert between types that are not directly compatible (e.g., string to int) or need more control over the conversion.

**Parse Method**:

* **What it is**: Used to convert a string into a specific data type, such as int.Parse() or double.Parse().
* **When to use**: Specifically for converting strings to numeric types.