إيه الفرق بين compiled و interpreted languages؟

Compiled Language:
اللغات دي بتتحوّل كلها مرّة واحدة لـ native machine code قبل ما تتنفّذ، عن طريق compiler.
يعني بيكون في ملف جاهز للتشغيل زي .exe، والمفيش حاجة بتحصل وقت التشغيل غير تنفيذ الكود.

أمثلة: C, C++
المميزات: أسرع وقت التشغيل، بس أقل مرونة وصعب تتبع الأخطاء.

Interpreted Language:
اللغات دي بتتنفّذ سطر بسطر عن طريق interpreter.
يعني كل سطر بيتقري ويتنفّذ في نفس اللحظة، ومفيش ملف جاهز زى الـ compiled.

أمثلة: Python, JavaScript
المميزات: أسهل في التجربة والتصحيح، بس الأداء أبطأ شوية.

طيب C# تبع أنهي نوع؟

الجميل في C# إنها hybrid، يعني بتجمع بين الاتنين:

أول حاجة الكود بيتحوّل لـ MSIL (Microsoft Intermediate Language) عن طريق compiler.

بعد كده، وقت التشغيل، بييجي دور الـ JIT (Just-In-Time) compiler اللي بيحوّل الكود لـ machine code ويشغله.

يعني C# مش 100% interpreted ولا 100% compiled، لكن بتعدي على مرحلتين علشان تستفيد من مميزات النوعين.

Implicit Casting:

بيحصل تلقائي من غير ما تكتب حاجة.

لما تحاول تخزن قيمة صغيرة في نوع أكبر (زي int في double).

ما فيش أي خطر أو فقدان بيانات.

```
int x = 10;
double y = x; // بيحصل لوحده
```

Explicit Casting:

لازم تكتبه بإيدك علشان ممكن يحصل فقد في البيانات.

بتحول من نوع أكبر لصغير، أو نوع مختلف.

```
double a = 9.8;
```

هنا هيفقد الكسور // int b = (int)a;

Convert:

طريقة جاهزة في C#، بتحاول تحوّل بين أنواع مختلفة.

لو القيمة null بيتعامل معاها بشكل آمن.

```
string s = "123";
int num = Convert.ToInt32(s);
```

Parse:

بيحول string لأرقام، لكن لازم القيمة تكون صح 100%.

لو فاضية أو مش رقم → Error.

```
string s = "123";
int num = int.Parse(s);
```