ماهو الClass :
هو عباره عن Temlpate قالب.

ما هو الObject:
هو عباره عن نسخه من ال Class

Complie Error:

هو خطا يظهر اثناء كتابة الكود قبل الRun

Run Time Error:

هو خطا يظهر بعد ال Run ويكون خطا بنتيجة الكود Output

Seald Class:

ما بتورث ولكن يمكن عمل Object منها

ما هي البرمجة الكائنية (OOP)؟
OOP هي **نمط برمجي** يعتمد على **الكائنات (Objects)** لتنظيم الكود، وتتكون من **أربع مبادئ رئيسية**:

- **التغليف (Encapsulation)**
- **الوراثة (Inheritance)**
- **تعدد الأشكال (Polymorphism)**
- **التجريد (Abstraction)**

ما الفرق بين الكلاس (Class) والكائن (Object)؟
✅ **Class** هو **القالب أو المخطط** لإنشاء الكائنات.
✅ **Object** هو **نسخة حية** من الكلاس تحتوي على **البيانات والوظائف**.

ما هو الفرق بين `Abstract Class` و `Interface`؟
✅ **Abstract Class**: يمكن أن تحتوي على **دوال عادية** ودوال مجردة (**مع كود وبدون كود**).
✅ **Interface**: تحتوي فقط على **تعريف الدوال** دون أي كود تنفيذ.

ما هو Polymorphism (تعدد الأشكال)؟
✅ يسمح باستخدام **نفس الدالة بأشكال مختلفة**، مثل **Overloading** و **Overriding**.

الفرق	Overloading	Overriding
التعريف	تعريف دالة بنفس الاسم ولكن **بمعلمات مختلفة**	إعادة تعريف دالة من **الكلاس الأب** في **الكلاس الابن**
التنفيذ	يتم في **نفس الكلاس**	يتم في **كلاس الوريث (Child Class)**
الكلمة المفتاحية	لا يحتاج إلى `virtual` أو `override`	يستخدم `virtual` في الأب و `override` في الابن

ما الفرق بين Encapsulation و Abstraction؟
✅ **Encapsulation (التغليف)**: إخفاء التفاصيل الداخلية للكود باستخدام **Modifiers مثل private و public**.
✅ **Abstraction (التجريد)**: إظهار **ما يهم فقط** وإخفاء التعقيد باستخدام **Abstract Classes و Interfaces**.

ما الفرق بين Composition و Inheritance؟
✅ **Inheritance (الوراثة)**: كلاس يرث من كلاس آخر.
✅ **Composition (التكوين)**: كلاس يحتوي على **كائن من كلاس آخر** بدلًا من الوراثة.

مثال من الحياة الواقعية على OOP

تخيل أنك في **شركة سيارات**، حيث يتم تصنيع أنواع مختلفة من السيارات مثل **Toyota و BMW و Ford**.

- **Class (الكلاس)** ⟵ هو المخطط الأساسي لأي سيارة.
- **Object (الكائن)** ⟵ كل سيارة يتم تصنيعها من المخطط.
- **Encapsulation (التغليف)** ⟵ المحرك مخفي داخل السيارة ولا يمكنك التحكم به مباشرة، بل عبر دواسة البنزين.
- **Inheritance (الوراثة)** ⟵ جميع السيارات تشترك في صفات معينة مثل العجلات والمحرك، لكن كل سيارة لديها ميزاتها الخاصة.
- **Polymorphism (تعدد الأشكال)** ⟵ جميع السيارات بها ميزة القيادة، لكن كل سيارة قد تتحرك بطريقة مختلفة (عادي، رياضي، كهربائي).

كيف تشرح OOP لطفل صغير؟ 👶

💡 مثال: لعبة ليغو (LEGO)

- **Class** ← هو تصميم المكعبات (القالب الأساسي).
- **Object** ← كل مكعب تبنيه هو كائن مختلف.
- **Encapsulation** ← لا يمكنك رؤية القطع الداخلية للروبوت، فقط يمكنك الضغط على زر التشغيل.
- **Inheritance** ← لديك سيارة LEGO، ثم تصنع منها سيارة سباق باستخدام نفس القواعد الأساسية.
- **Polymorphism** ← يمكنك تشغيل الروبوت بطريقة **سريعة أو بطيئة** حسب الزر الذي تضغط عليه.

كيف تشرح OOP لرجل كبير في العمر؟ 👴

💡 مثال: الأسرة والعائلة

- **Class (الكلاس)** ← "العائلة" هي المفهوم الأساسي الذي يحتوي على صفات مشتركة مثل الاسم والعمر.
- **Object (الكائن)** ← "أنت" و "أولادك" هم أفراد العائلة، كل واحد كائن مستقل لكنه يتبع نفس القواعد.
- **Encapsulation (التغليف)** ← مثل خزنة العائلة، بعض الأمور لا يستطيع أي شخص خارج العائلة الوصول إليها.
- **Inheritance (الوراثة)** ← الأبناء يرثون الصفات من الأبوين، مثل لون العيون والطول.
- **Polymorphism (تعدد الأشكال)** ← كل فرد بالعائلة يتحدث بطريقة مختلفة، لكن الجميع يستخدم نفس اللغة الأساسية.