OOP prinsipləri

OOP-nin 4 əsas prinsipi vardır:

1. Abstraction

2. Encapsulation

3. Ploymorphism

4. İnheritance

İnheritance. Bir obyektin başqa bir obyektdən törədilə bilməsi fikridir. Belə ki, bu xüsusiyyət proqramistə oxşar xüsusiyyətləri olan obyektləri dəfələrlə təkrar elan etmədən yarada bilmək imkanı verir. Buna misal olaraq “nəqliyyat vasitələri” sinfindən “velosiped”, “avtomobil” siniflərinin törəməsini göstərə bilərik.

Encapsulation. Metodlar vasitəsilə obyektə aid məlumatları gizləmək rolunu oynayır. Həmin məlumatları gizləmək üçün dəyişənləri elan edən zaman “access modifiers” adlanan təyinedicilərdən istifadə edilir. Bu təyinedicilər dəyişənlərin başqa klaslarda görünənliyini əlçatanlığını müəyyən etmək üçün istifadə olunur. Bunlara aşağıdakılar aiddir: *public, private, protected*

Public- elan olunan dəyişən həmin metod daxilində bütün klaslarda istifadə edilə bilər.

Private- dəyişən yalnız elan olunduğu klasın tərkibində çağırıla bilər.

Protected- dəyişən elan olunduğu klas və ondan törəyən klaslarda çağırıla bilər.

Abstraction. Obyekt haqqında verilən informasiyanın yalnız vacib olan hissəsinin göstərilib, qalanının isə gizlədilməsi üçündür. Bunu real misal üzərində belə izah edə bilərik ki, maşında sürücünün görüb bildiyi hissələr yalnız onun bəzi xüsusiyyətləridir, məs: rəngi, necə işə salınması və s. Lakin açarı daxil edib mühərriki işə salan zaman arxada hansı proseslərin getməsi isə gözlə görünmür. Abstraction da belədir, burada nəyin istifadəçi tərəfində görünüb görünməyəcəyini özümüz təyin edirik.

Abstract class-lar əsasən ortaq xüsusiyyətləri olan klaslar üçün “ana klas” rolunu oynayır. Bunlar kod təkrarını azaltmaq üçün istifadə olunur və bu klaslardan “new” açar sözü ilə obyekt yarada bilmərik.

Polymorphism. Bu özəlliyi bir metodun fərqli klaslarda fərqli funksiyalar yerinə yetirə bilməsi kimi açıqlaya bilərik. Yəni, bir metod təyin edilir həmin metod eyni bir sinifdən törəyən “övlad” siniflərin hər birində fərqli cür nəticə verə bilər. Buna çox kiçik bir misal üzərində baxaq. Məsələn bizim “ANİMALS” adlı bir klasımız və ondan törəyən “DOGS”, “CATS” klaslarımız var. Bildiyimiz kimi bu klasların hər ikisi birinci klasımızdan törəyir, lakin onların ümumi xüsusiyyətlərindən əlavə fərqli xüsusiyyətləri də var, məsələn hər biri fərqli səslər çıxarır. Buna görə də bir “makeSounds()” metodu təyin etsək, hər bir klas daxilində onu çağıranda fərqli dəyərlər ötürdüyümüz üçün nəticələr də fərqli olacaq. Bu da bizə polimorfizmi başa düşməyə kömək edir.

OVERRIDE. Bir sinifə aid metodun həmin klasdan törədilən başqa bir klas içərisində eyni adlı metod yaradılaraq “ana” klas olan metodun əvəz olunması kimi izah oluna bilər. Burada diqqət edilməsi gərəkən bir məqam var ki, bu metodlar “private” elan olunmamalıdır. Bir metodu “override” edə bilməmiz üçün isə əvvəlcə onu virtual olaraq tanıtmalıyıq. Yəni bir neçə klasımızın eyni məna verən metodları varsa bu metodu “virtual” olaraq tanıdırıq.

**BİLMİRƏM NƏ DƏRƏCƏDƏ DÜZGÜN İFADƏ EDƏ BİLDİM.....**