**Obyektyönlü proqramlasdırma**

*Obyekt* anlayışı proqramlaşdırmada mühüm rol oynayır. Proqram obyektləri real həyatdakı obyektlərə çox bənzəyir – onlar da öz daxili quruluşları, hərəkətləri, xassələri ilə bir-birindən fərqlənir. Kompüterin ekranındakı hər bir şeyə obyekt kimi baxmaq olar: ekranda istənilən pəncərə obyektdir; hər bir düymə, simgə obyektdir; istənilən menyu obyektdir. İlk növbədə koddan təkrar istifadə prinsipi yaranmışdı. Bu prinsipə görə, kiminsə bir dəfə yaratmış olduqları toplanıb saxlanmalı və başqa proqramçılar tərəfindən hazır bloklar şəklində istifadə olunmalıdır. Belə proqram bloklarını obyektlər (OBJECT) adlandırdılar. Yeni proqram hazırlamaq lazım gəldikdə, əvvəlki proqramlardan obyektlər götürülür və onlar, sadəcə, yeni tələblərə uyğun dəyişdirilir. Müxtəlif proqramlarda eyni formalı pəncərələrə, oxşar menyulara, eyni düymələrə rast gəlinir. Proqramların əksəriy-yətində eyni şriftlərədən istifadə olunur. Bunların hamısı yeni tələblərə uyğunlaşdırılmış obyektlərdir.

Object Oriented Programming (Obyekt Yönümlü Proqramlaşdırma) meydana gəlməmişdən əvvəl proqramçılar aşağıdakı problemlərlə qarşılarşırdılar.

1. Proqramçının yazdığı kod hissələrə bölünə bilmədiyi üçün,o , kodun hər bir sətrinə bələd olmalı idi;
2. Proqrama yeni kodlar əlavə etmək, tamamilə, problem idi.Çünki, bu kodun

proqrama uyğunlaşması çox vaxt tələb edirdi;

1960-cı illərdə, bu problemlərə qarşı duran OOP (Object Oriented Programming) meydana gəldi. Proqramlaşdırmadakı obyektlərin eyni və ya bənzər xüsusiyyətlərinə əsaslanan model və ya paradiqmadır. Obyekt Yönümlü Proqramlaşdırmada kod obyektin atributları (atributes) və funksiyalarını (functions) nəzərə alınaraq sinifləndirilir. Məsələn, rəngləri müxtəlif olan qələmlər ixrac edilməlidir. Buradakı obyekt qələmdir, atributu rəngi, funksiyası isə yazmaqdır.

**Obyekt nədir?**

1. Obyekt aid olduğu sinifdə xüsusiyyətləri, yəni, atributları, funksiyaları olan və

RAM`da yer tutan varlıqdır.

**Class (Sinif) nədir?**

1. Class fərqli atributlərı olan və fərqli funksiyalar yerinə yetirən obyektlərin ortaq xüsusiyyətlərinin əks etdirən (declare) yerdir. OOP`nin əsas prinsipi kodun sistemli şəkildə sinifləndirilmiş olmasıdır.

**Constructor (Konstruktor) nədir?**

1. Constructor, obyekt yaradıldıqda işə düşən və obyektin atributlarını, funksiyalarını tərkibində saxlayan metoddur (initialization method).

***Obyekt Yönümlü Proqramlaşdırmanın 4 əsas prinsipi var:***

1. ***İnheritance (miras alma) —*** Müəyyən bir class`dan özü əsasında yeni class

yaradılır. Yəni, əsas class`ın xarakteristikaları digər class`lara miras verilir. Belə classların 2 növü var: Child class və ya sub-class (alt sinif) və Parent class və ya super class (üst sinif). İnheritance vasitəsilə kodda yersiz təkrarın qarşısı alınır.

1. ***Polymorphism (Polimorfizm)* —** Yaratdığımız obyektin özünü lazım olanda,sanki , başqa bir obyekt kimi aparmasına deyilir. Yəni, eyni obyektin fərqli səbəblər üçün istifadəsini təmsil edir.

***Polymorphism və İnheritance arasındakı fərqləri aşağıdakı kimi yaza bilərik:***

1. *İnheritance* — Super Class`dakı xüsusiyyətlərin yeni Classda işlənməsini təmin edir, *Polymorphism* — Eyni obyekti müxtəlif üsullarda işlətməyə imkan verir.
2. *İnheritance* — class səyyəsində tətbiq edilir, *Polymorphism* — metod səviyyəsində tətbiq edilir.
3. *İnheritance —* kodun yenidən istifadə edilməsini təmin edir, *Polymorphism* —

kod yazılarkən və işə düşərkən obyekti və ya metodu çağırmağa imkan verir.

***Abstraction (Abstraktlaşma) —*** Bir Class`ın daxilində olan mürəkkəb, qarışıqlığı artıran hissələrin digər Class`da işlədilməməsidir. Beləliklə, bu Child Class`da mürəkkəb funksiyaların necə yerinə yetirildiyini görə bilmirik.

***Encapsulation (Kapsullaşdırma)*** — Bir Class`da verilən məlumatlara digər Class`lardan əlçatanlığın idarə edilməsidir. Bu məlumatlar public, private, protected

adlandırılan terminlərlə (access modifiers) idarə olunur. *Public* kimi təyin etdiyimiz məlumat qalan bütün siniflər üçün əlçatandır. *Private* kimi təyin etdiyimiz məlumat sadəcə həmin Class üçün əlçatandır, digər Class`lar bu məlumatı əldə edə bilmir. *Protected* ilə təyin etdiyimiz izə bu Class`dan miras alan (inherit) digər Class`lar üçün əlçatandır.

***Abstraction və Encapsulation arasındakı fərqləri aşağıdakı kimi yaza bilərik:***

1. *Abstraction* — lazım olmayan məlumatları gizlətmək üçün istifadə

olunur, *Encaptulation* — məlumatın gizliliyini idarə etmək üçün istifadə olunur.

1. *Abstraction* — dizayn səviyyəsində tətbiq edilir, *Encaptulation* — tətbiq səviyyəsində tətbiq edilir.
2. *Abstraction* — əsasən, obyektin xarici xüsusiyyətləti üçün istifadə olunur, məsələn, mobil telefonun ölçüsü, rəngi və s. , *Encaptulation* — obyektin daxili

xüsusiyyətləri üzərində tətbiq olunur, məsələn, mobil telefonun düymələrinin ekran ilə əlaqəsi və s.

Obyektyönlü proqramlaşdırmanın bir çox vasitələri **Simula-67** dilindən götürülmüşdür. Bu dil 1970 -1980 -ci illərdə yaradılmışdır. Proqramda obyektlərin təsviri, məcmuları, onların arasında əlaqələr və qarşılıqlı təsirinin üsulları izah edilmişdir.

Proqramlaşdımanın **obyektyönlü** üslubu obyekt anlayışına əsaslanır, mənası isə **«obyekt - verilənlər + prosedurlar»** düsturu ilə ifadə olunur. Hər bir obyekt verilənlərin strukturunu birləşdirir və onlara müraciət bu verilənbrin email proseduru ilə mümkündür ki, bu da metod adlanır.

Click "Allow" to open the full version of the site