Proqramlaşdırma Paradigmaları

İmperativ programlaşdırma

Bir programlaşdırma dilindəki əmrlərin sətir-bə-sətir verilməsidir. Məsələn bir robota əmrlər versəydik:

qolu 10 dərəcə sağa döndər

2 metr irəli hərəkət et

- *qolu 20 dərəcə sola qaldır*

və s. bu kimi əməliyyatları sıralamamız lazım olacaq, bu əməliyyatlar qolu hərəkət etmə əmrinin alt əmrləridir.

Bu əmirlərə prosedur da deyilir. Əmrli programlaşdırmada önəmli olan hesablamanın "necə?" aparılacağıdır. Ona görə də maşın dilinə ən yaxın olan paradigma "İmperativ" likdir. Məsələn siz tez-tez programlaşdırma öyrənməyə "C" dən başla kimi bir ifadə eşitmiş olarsınız. Bunun səbəbi də C nin imperativ bir dil olmasındadır. Bu xüsusiyyət C ni olduqca çətin mənimsənilən bir dil halına gətirir. C də yazan bir programçı funksionallıqdan istifadə etmədiyi üçün, bütün programı əmrlərlə yazdığı üçün programın ən çətin hissəsini yazmış olur. Və ona görə də C developerlərin başqa dili öyrənmələri asanlaşır.

Object Oriented Programming

- *OOP 1960- cı illərdən bu günə qədər programlaşdırma dünyasında öz töhvəsini vermiş bir paradigmadır. Belə ki 1970 lərdən bu günədək qurulan programlaşdırma dillərindən bir çoxunda OOP məntiqi istifadə olunur. Məsələn hal-hazırda ən məşhur olan "Python,Java,C#" kimi dillərdə istifadə olunur. OOP qurulduğu andan özündən əvvəlki programlaşdırma məntiqini tamamilə,kökdən dəyişdirmişdir. OOP dən əvvəlki metodologiya "Prosedural Programlaşdırma" adlanırdı. Bu metodologiya uzun zaman aktiv olsa da bir çox mənfi cəhəti var idi. Məsələn yazılan programları parçalamaq mümkün deyildi, programlar tam halda olurdu ki, bu da programa hər hansı dəyişiklik edilməsini çətinləşdirirdi. Beləliklə çox kiçik səhvlər belə böyük bir problemin yaranmasına gətirib çıxarırdı. Məhz bu problemi aradan qaldırmaq üçün OOP məntiqi yaradıldı. Bu məntiqi ilk dəfə ortaya atan **Alan Kay** bunu belə izah edir:*
- *Program, obyektlər və onların sərhədləri çərçivəsində müəyyən bir işi yerinə yetirmək üçün yazılmalıdır. Hər obyektin bir sinifi(class) olmalıdır və bu siniflər obyektlərin eyni tərzli davranışlarını ifadə etməlidir. Obyektlər bir-biri ilə əlaqəli ola bilməlidirlər.*
- *Bu paradigmanın ən müsbət tərəfi yazacağımız sinifləri bir-birindən asılı olmadan genişləndirə bilməyimizdi. Bunun sayəsində program "Böl,Parçala və Fəth Et" məntiqi çərçivəsində kiçik hissələrə parçalanaraq, ayrı-ayrı hissələrlə ifadə oluna bilir. OOP məntiqində real dünyadakı obyektlər nümunə alındığı üçün kodda daha rahat şəkildə başa düşülür.*

Parallel Programming

- *Paralel programlaşdırma böyük əmrlərin və ya böyük problemlərin hissələrə bölünərək həll edilməsidir. Paralel paradigma sayəsində bir əmri bir neçə hissəyə bölüb ən əsasıda böldüyümüz hissələri eyni anda işə salmağa müvəffəq oluruq. Əgər Paralel programlaşdırma ilə etməsəydik, bu böyük əmri bir dəfəyə verdiyimiz və ya hər hansı böyük bir əməliyyatı hissələrə bölmədən bütöv bir şəkildə icra etdiyimiz üçün komputer bu tək və böyük əmri çətinliklə yerinə yetirəcəkdi. Lakin, əmri hissələrə bölüb "hadi koçum, yap şimdi " dediyimiz anda komputer bu bölünmüş bəsit əmrləri asanlıqla yerinə yetirir və təkrar edirəm, ən əsasıda bir neçə əmri eyni anda sıraya qoymadan yerinə yetirməklə bizə vaxt qazandırır. Sizin dərsdə çəkdiyiniz misala bənzəyir bu. Siz videonu 1 komputerdə save və ya convert etdiyinizə görə, komputer əməliyyatın təxmini bitmə müddətinin 1-2 həftə olduğunu hesablayıb. Lakin, siz həmin videonu göndərmisiniz almaniyada bir neçə ayrı komputerdə daha sürətli bir şəkildə etmisiniz.*

-

Deklarativ Proqramlaşdırma

- *Deklarativ proqramlaşdırma real dünya ssenarisi ilə izah edilə bilər. İstifadəçinin yeni e -poçtları yoxlamalı olduğunu düşünün. Bir üsul, gələnlər bildirişlərini aktivləşdirməkdir. İstifadəçi bildirişləri yalnız bir dəfə aktivləşdirməlidir və hər dəfə yeni bir e -poçt gəldikdə avtomatik olaraq bildiriş alır. Deklarativ proqramlaşdırma buna bənzəyir. Sadəlik təmin edir. Deklarativ proqramlaşdırma tələb olunan nəticənin nə olduğunu ifadə edir. Nəzarət axınını təsvir etmədən hesablamanın məntiqini izah edir.*

Funksional paradigma

Functional Programming*(Funksional programlaşdırma) hal-hazırda mövcud olan programlaşdırma dillərinin çoxunda istifadə olunan yanaşmadır. Funksionallıq sadə dillə desək:*

— Kod təkrarının qarşısını alır və eyni kodun fərqli şərtlərdə təkrar istifadəsini

təmin edir.

— Kodun oxuna bilinməsini artırır və kodun analizini sadələşdirir.

— Programın hazırlanma mərhələsində programçıya modullarla işləməsini təmin edir.

Bir dilin funksional olması bu dildə funksiya və ya prosedur tipli xüsusiyyətlərin olması ilə xarakterizə olunur. Aşağıdakı kod nümunəsi ilə funksionlığı daha yaxşı başa düşəcəksiniz:

```
int topla ( int a, int b){
    return a + b;
    }
```

*Bu kodda a və b funksiyaya verilən parametrlərdir, dönüş dəyəri olaraq tam ədəd(integer) istifadə olunub və bu hal **return** əmri ilə ifadə olunub, "a+b" əməliyyatı isə bir alt programdır. Yəni **topla** funksiyası çağırıldığında icra olunur.*

Logic programming

- *Məntiq proqramlaşdırma əsasən formal məntiqə əsaslanan proqramlaşdırma paradiqmasıdır. Məntiqi proqramlaşdırma dilində yazılan hər hansı proqram hansısa problem sahəsinə aid faktları və qaydaları ifadə edən məntiqi formada cümlələr toplusudur. Əsas məntiq proqramlaşdırma dil ailələrinə Proloq, cavab dəsti proqramlaşdırması (ASP) və Datalog daxildir. Bu dillərin hamısında qaydalar cümlələr şəklində yazılır:*
-
- *H :- B1, …, Bn.*
- *və məntiqi nəticələr kimi bəyannamə ilə oxunur:*
-
- *H, əgər B1 və … və Bn.*
- *H qaydanın başı, B1, ..., Bn isə gövdə adlanır. Faktlar gövdəsi olmayan və sadələşdirilmiş formada yazılmış qaydalardır:*
-
- *H.*
- *H, B1, ..., Bn-nin hamısı atom düsturları olduğu ən sadə halda, bu bəndlər müəyyən cümlələr və ya Horn cümlələri adlanır. Bununla belə, bu sadə halın bir çox uzantıları var, ən vacibi, bəndin gövdəsindəki şərtlərin də atom düsturlarının inkarı ola biləcəyi haldır. Bu genişlənməni özündə birləşdirən məntiq proqramlaşdırma dilləri monoton olmayan məntiqin bilik təmsil imkanlarına malikdir.*