Polimorfizm - çoxşəkillilik deməkdir

Bir obyektin funksiyasinin ozunu basqa obyektlerde ferqli aparmasi polimorfizmdir.

Mes. boyukle basqa cur danisirsan, 5 yasli usaqla basqa

Polimorfizmin olmasi runtime zamani bas verir

Inheritance vasitesiyle base classin pointerinde o ve onun toremelerinin tiplerini saxlaya biler;

Yeni :

Animal\* animal = new Car(); yaza bilerik

(ozunden gelen,torenen tip)

Inheritancein verdiyi mohtesemlik:

Inheritance vasitesiyle bir massivin icinde muxtelif tipli, lakin eyni basse classdan torenmis elementler saxlaya bilerik

///////////////////////////////////////////////////////////////////////////////////////////////

virtual acar sozu ile override edirik (yeni metodu basqa obyekte gore muxtelif cur reallasdirdi);

Meselen;

class Base

virtual string GetName()

{

return "I am base";

}

class Derived :public Base

string GetName()

{

return "I am derived";

}

main()

Base b(10);

Base\* base = new Derived(20);

cout << base->GetName() << endl;

OUTPUT:

I am derived

///////////////////////////////////////////////////////////////////////////////////////////////

//\*Metodlar tamamile eyni olmalidir((const)//

///////////////////////////////////////////////////////////////////////////////////////////////

Klasslar arasi dinamik polimorfizmin olmasi ucun base classa pointer saxlamaliyiq

Hansi metodun muxtelif olmagini isteyirikse qarsisina virtual yaziriq

Inheritancein bize verdiyi komekliklerden biri:

base pointer saxlamaqla basein torenenlerini massivde saxlaya bilerik

///////////////////////////////////////////////////////////////////////////////////////////////

1)Early binding or Static Polymorphism :

Statik polimorfizmdir, kompliyasiya vaxtinda neyi cagiracagini bilir

Mes.

void Show()

{

cout << 10 << endl;

}

main()

{

Show();

}

2) Late binding or Dynamic Polymorphism :

program run olandan sonra, run timedan sonra neyi cagiracagini bilir

Mes.

void add(int a, int b) {

cout << a + b << endl;

}

main()

{

void(\*ptr)(int, int) = add;

ptr(10,20);

}

///////////////////////////////////////////////////////////////////////////////////////////////

Polimorfizmin olmasi ucun inheritance lazimdir mi?

Yox, cunki funksiyaya pointer saxlamaqla late binding ede bilerik

///////////////////////////////////////////////////////////////////////////////////////////////

Torenen klasslara overrride ederken, hemin overridein tamamile dogru oldugun qaranti etmek ucun hemin methodun

qarsisindan override acar sozunu yaza bilerik (yazmasaqda override edir lakin qaranti etmek ucun, hemcinin oxunarligini artirmaq ucun yazila biler)

///////////////////////////////////////////////////////////////////////////////////////////////

A (virtual)

B:A

C:B

virtualliq(override) C-ye qeder (sona kimi) torenir ve A dan baslayir

///////////////////////////////////////////////////////////////////////////////////////////////

A (virtual)

B:A (final) (sintaksis - void Speak() override final)

C:B

final keyword oldugu ucun virtualliq(override) B ye qeder torenir

///////////////////////////////////////////////////////////////////////////////////////////////