ILM FAN XABARNOMASI
Ilmiy elektron jurnali

OBYEKTGA YO'NALTIRILGAN DASTURLASH

Erkinov Azizbek
Toshkent Davlat Iqtisodiyot Universiteti
Azizbekerkinov16@gmail.com
Alisher Ismailov
Toshkent Davlat Iqtisodiyot Universiteti
Email

Annotatsiya: Ushbu maqolada men "Obyektga yo'naltirilgan dasturlashning asosiy tushunchalarini" yoritib beraman.Bu maqolada obyektga yo'naltirilgan dasturlash (OOP) paradigmasining asosiy tushunchalari, tamoyillari va ularning dasturlashda qo'llanilishi keltirilgan. OOP dasturlashda ma'lumotlarni va ularni boshqaruvchi funksiyalarni birlashtirishga yordam beruvchi modeldir, bu esa kodni soddalashtirish va uni boshqarish imkoniyatlarini oshiradi.Maqolada OOP ning asosiy tushunchalari: obyekt, sinf, inkapsulatsiya, meros va polimorfizm kabi atamalarning ta'rifi berilgan. Har bir tushuncha alohida misollar orqali tushuntirilgan, bu o'quvchilarga OOPni chuqurroq tushunishga yordam beradi. OOP dasturlashda modullilikni oshiradi, qayta ishlatish imkoniyatlarini kengaytiradi va xatoliklarni kamaytirishga yordam beradi.
Maqola shuningdek OOP ning dasturlash tillaridagi C++ roli va ulardagi qo'llanilishini ko'rib chiqadi.OOP tizimlarining o'zaro bog'liqligi va ularning zamonaviy dasturlashdagi ahamiyati haqida ham so'z yuritiladi.Ushbu maqola OOP ning dasturlashni qanday rivojlantirganini va undan qanday foydalanishni tushunishga qaratilgan bo'lib, dasturlashni o'rganayotganlar uchun muhim ma'lumotlar beradi.

Kalit so'zlar: obyekt,polymorfizm,abstract,sinflar,inkapsulyatsiya,class,dasturlash paradigmalari,meros olish

Kirish

Dasturlash paradigmalari - bu dasturiy ta'minot ishlab chiqishda asosiy yondashuvlarni ifodalaydigan tamoyillar va usullar to'plamidir. Dasturchilar har bir paradigma yordamida tizimni yaratishda ma'lum bir usulni tanlaydilar. Obyektga Yo'naltirilgan Dasturlash (OOP) paradigmasi dasturlashda eng ommabop yondashuvlardan biri bo'lib, u kodni tuzish va tizimlarni tashkil etishning samarali usulini taklif etadi. OOP dasturlashning asosiy tamoyili shundaki, tizimdagi barcha narsalar - obyektlar orqali ifodalanadi. Bu yondashuv ma'lumotlar va ular bilan ishlaydigan metodlarni birlashtirib, tizimni ancha tushunarli va boshqariladigan qiladi.
OOP asosida tizimni yaratishning foydalari katta: tizimlar modullarga bo'linadi, ya'ni har bir modul o'zining mustaqil funksiyalarini bajaradi, bu esa dasturiy ta'minotni rivojlantirish va test qilishni osonlashtiradi. Bundan tashqari, OOP yordamida kodni qayta ishlatish (reusability) mumkin, bu esa kodni yozish va texnik xizmat ko'rsatish jarayonini sezilarli darajada tezlashtiradi.

OOP metodologiyasi 1960-yillarda Shvetsiya olimi Ole-Johan Dahl va Kristen Nygaard tomonidan yaratilgan. Ular ob'ektga yo'naltirilgan dasturlashni kompyuter tizimlarini simulyatsiya qilishda qo'llashgan va bu yondashuv dasturlashda yangi bir sahifani ochdi. Hozirgi kunda OOP ko'plab zamonaviy dasturlash tillarida (masalan, Java, C++ Python, C#, Ruby) keng qo'llaniladi va ular bu paradigmaning eng yirik tarafdorlari hisoblanadi. OOP yordamida ishlab chiqilgan dasturlar modular tuzilishga ega bo'ladi, ya'ni har bir modul o'ziga xos vazifalarni bajaradi va mustaqil ishlaydi. Bu, dasturiy ta'minotni rivojlantirishda yuqori darajadagi samaradorlikka olib

keladi. OOP usuli orqali ob'ektlar va sinflar o'rtasidagi munosabatlar, ya'ni ob'ektlarni boshqarish, ularning holatini saqlash va metodlar yordamida ularni manipulyatsiya qilish ancha osonlashadi.OOP paradigmasining ko'p afzalliklaridan biri - bu kengaytiriluvchanlik (extendibility). OOP yordamida yaratilgan tizimlarni yangi xususiyatlar va funksiyalar bilan oson kengaytirish mumkin, bu esa vaqt o‘tishi bilan dasturga yangi imkoniyatlarni qo‘shishni ta'minlaydi.

Shu tariqa OOP nafaqat dasturlashni yanada samarali qilish, balki tizimlarni boshqarish va rivojlantirishni ham osonlashtiradi. OOPni chuqurroq o'rganish, dasturchilarni nafaqat ma'lum bir dasturlash tilini o‘rganishga, balki umumiy dasturlash amaliyotiga yuqori darajada egallashga ham yordam beradi.

OOPning asosiy tushunchalari

Obyekt - dasturdagi asosiy qurilish blokidir. Bu ma’lumotlarni va ularga tegishli metodlarni (yoki amallarni) o'z ichiga olgan tuzilmadir. Ob'ektlar sinf (class) asosida yaratiladi va ular sinfda belgilangan atributlar (ma'lumotlar) va metodlar (funksiyalar) yordamida ishlaydi.Obyektlar dasturda o'zgaruvchi qiymatlar yoki amallarni o'z ichiga olgan mustaqil birliklar sifatida ishlaydi. Har bir ob'ekt o'ziga xos xususiyatlarga ega bo'ladi, masalan, avtomobilning modeli, ranglari, tezligi, va shu kabi boshqa atributlar va metodlar bilan ishlash.

Sinf - bu obyektlarni yaratish uchun shablon yoki tavsifdir. Sinf obyektlarning umumiy xususiyatlarini ta'riflaydi va ular asosida konkret ob'ektlar yaratiladi. Sinf ichida atributlar (ob'ektning xususiyatlari) va metodlar (obyektga tegishli amallar) belgilangan bo'ladi.Sinf obyektlarning xususiyatlari va amallarini umumiy shaklda ta'riflashga imkon beradi, bu esa kodni soddalashtiradi va tizimni boshqarishni osonlashtiradi. Har bir yangi obyekt sinfga asoslanib yaratiladi va shu asosda ishlaydi.

Inkapsulyatsiya - bu ob'ektning ma'lumotlarini (atributlarini) tashqi dunyodan yashirish va ularga faqat metodlar orqali kirishni ta'minlash jarayonidir. Inkapsulyatsiya tizimdagi ma'lumotlarning yaxlitligini ta'minlaydi, chunki tashqi kod obyektning ichki holatini bevosita o'zgartira olmaydi.Bu yondashuv obyektga faqat zarur metodlar orqali murojaat qilishni ta'minlab, tizimning xavfsizligini oshiradi. Inkapsulyatsiya orqali ob'ektlar bilan ishlashni soddalashtirish va ma'lumotlarni himoya qilish mumkin bo'ladi.

Meros - bu yangi sinfning boshqa sinfdan xususiyatlar va metodlarni meros qilib olishidir. Bu ob'ektlar o'rtasida umumiy xususiyatlar va metodlarni qayta ishlatish imkonini beradi va kodni optimallashtirishga yordam beradi.Meros orqali sinflar o'rtasida ierarxiya (otasi va farzandi) yaratiladi. Yangi sinf avvalgi sinfning atributlari va metodlarini avtomatik ravishda o'zlashtiradi, bu esa kodning qayta ishlatilishini ta'minlaydi. Bu yondashuv tizimni kengaytirishda va uni boshqarishda yordam beradi.

Polimorfizm — bu bir xil metodning turli ob'ektlar uchun turlicha ishlashidir. Ya'ni, bir metod bir necha turdagi obyektlar ustida ishlashi mumkin va har bir obyekt uchun turli natijalar beradi. Bu imkoniyat OOPda kodning qayta ishlatilishini va moslashuvchanligini ta'minlaydi.Polimorfizm dasturlashda metodlarning turli obyektlar uchun moslashuvchan tarzda ishlashiga imkon beradi, bu esa tizimning kengaytirilishini va yanada modul tuzilishini ta'minlaydi.

Abstraksiya - bu tizimdagi faqat kerakli ma'lumotlarni ko'rsatish va keraksiz detallardan yashirish jarayonidir. Abstraksiya orqali tizimni soddalashtirish mumkin bo'ladi, chunki foydalanuvchi faqat tizimning ishlashiga zarur bo'lgan interfeysni ko'radi va uning ichki ishlashini bilmasligi mumkin.Abstraksiya tizimning murakkab qismlarini yashirib, foydalanuvchiga faqat umumiy va zarur ma'lumotlarni taqdim etadi. Bu tizimni tushunishni osonlashtiradi va foydalanuvchi bilan tizimning o'zaro aloqasini soddalashtiradi.

OOP yutuqlari

Obyekt. Modullik: Har bir obyekt o'zining ma'lumotlari va metodlariga ega bo'lib, tizimni alohida qismlarga bo'lish imkoniyatini yaratadi. Bu, katta dasturlarda o'zgarishlar kiritishni osonlashtiradi va tizimning modulli tuzilishini ta'minlaydi.

Kengaytirilish: Obyektlar yangi xususiyatlar qo'shilishi orqali kengaytirilishi mumkin, bu esa dasturiy ta'minotning o'sishini va yangi talablar asosida o'zgarishlarni qo'llab-quvvatlaydi.

Sinf. Kodning qayta ishlatilishi: Sinflar yordamida kodni qayta ishlatish mumkin. Sinfda bir marta yozilgan xususiyatlar va metodlarni bir nechta ob'ektlar uchun ishlatish imkonini beradi. Bu, ishlab chiqish vaqtini qisqartiradi va kodni samarali boshqarishga yordam beradi.Strukturali tashkil etish: Sinflar dasturdagi kodni tuzish va tartibga solish imkonini beradi, bu esa kodni o'qish va tushunishni osonlashtiradi. Sinflar obyektlar o'rtasida o'xshashliklarni aniqlash va boshqarish uchun qulay.

Inkapsulyatsiya afzalligi. Ma'lumotlarning himoyasi: Inkapsulyatsiya ob'ektning ma'lumotlarini tashqi aralashuvlardan himoya qiladi. Ma'lumotlar faqat metodlar yordamida o'zgartirilishi mumkin, bu esa xatoliklarni kamaytiradi va tizimning xavfsizligini ta'minlaydi.Xatoliklarni kamaytirish: Ma'lumotlar va metodlarning ajratilishi, tizimdagi noaniqlik va noxush holatlarning oldini olishga yordam beradi. Dasturchi faqatgina kerakli metodlar orqali ob'ektga kirish imkoniyatiga ega bo'ladi.

Meros. Kodning qayta ishlatilishi: Meros yordamida mavjud sinflar asosida yangi sinflar yaratish mumkin. Bu, bir xil kodni bir nechta joyda yozishdan saqlaydi va tizimning modifikatsiyasini osonlashtiradi.Oson kengaytirilish: Yangi sinflar eski sinflardan meros qilib olish orqali ularni kengaytirishi va ularning xususiyatlarini o'ziga moslashtirishi mumkin. Bu dasturda yangi xususiyatlarni qo'shishni osonlashtiradi.

Polymorfizm. Moslashuvchanlik: Polimorfizm bir xil metodni turli obyektlar uchun ishlatishga imkon beradi. Bu bir nechta sinfga mansub obyektlar o'rtasida umumiy metodni qayta ishlatish imkoniyatini beradi, bu esa tizimni moslashuvchan va kengaytiriladigan qiladi.Kodning soddalashtirilishi: Turli sinflardan bir xil metoddan foydalanish orqali dasturchilar ko'proq samarali va sodda kod yozadilar. Polimorfizm tizimni yangi talablar asosida kengaytirishni osonlashtiradi.

Abstraktsiya. Tizimning soddalashtirilishi: Abstraksiya orqali tizimning faqat kerakli qismlari ko'rsatiladi, keraksiz tafsilotlar esa yashiriladi. Bu foydalanuvchi yoki boshqa dasturchilar uchun tizimni oson tushunishga yordam beradi.Ishlatish qulayligi: Abstraksiya interfeyslarni yaratish imkonini beradi, shunda foydalanuvchi faqatgina zarur bo'lgan xususiyatlar va metodlardan foydalanadi. Bu orqali dastur bilan ishlash qulaylashadi va tizimning ichki ishlashini bilmasdan foydalanish mumkin.

OOP dasturlashga misollar

```
#include <iostream>
using namespace std;
class Animal {
public:
    virtual void speak() = 0;
};
class Bird : public Animal {
public:
    void speak() override {
```

```
        cout << "Chir-chir!" << endl;
    }
};
class Fish : public Animal {
public:
    void speak() override {
        cout << "Bubble bubble!" << endl;
    }
};
int main() {
    Bird bird;
    Fish fish;
bird.speak();
fish.speak();
    return 0;
}
```

Ko’rib turganizdek abstrakt sinf yaratdik.

```
#include <iostream>
  using namespace std;
class Animal {
  public:
      void eat() {
cout << "Bu hayvon ovqatlanmoqda." << endl;
      }
  };
 class Dog : public Animal {
  public:
      void bark() {
          cout << "Bu kichik vovulamoqda." << endl;
      }
  };
 int main() {
      Dog myDog;
      myDog.eat();
      myDog.bark();
      return 0;
  }
```

Bunda biz meros orqali bitta sinfning xususiyatlarini boshqa sinfga o'tkazdik.Yani ot klassdan bola klassga o’tkazganimizdaqa.

Xulosa

OOP yordamida tizimlar oson kengaytiriladi, ma'lumotlar tashqi aralashuvlardan himoyalanadi va kodni boshqarish, xatoliklarni kamaytirish osonlashadi. OOP paradigmasi dasturchilarga

kompleks tizimlarni samarali va tushunarli tarzda yaratish imkonini beradi, shu bilan birga, kengaytirilgan va uzoq muddatli dasturlarni ishlab chiqishga imkon beradi.

Umuman olganda OOP - bu dasturiy ta'minotni yaratishda yuqori samaradorlik va yaxlitlikni ta'minlaydigan kuchli metodologiya bo'lib, bugungi kunda deyarli barcha zamonaviy dasturlash tillari va tizimlarida keng qo'llaniladi.

OOP ning kamchiliklari ham qisman mavjud – dasturlashga yangi qadam qo'yganlar uchun biroz tushunarsiz , Har doim ham samarali emas , misol uchun: chiziqli va kichik dasturlarda va bazida dasturimiz haddan tashqari.

Foydalanilgan adabiyotlar

G'aniyev_S_K,_Karimov_M_M,_Tashiyev_K_V_Axborot_xavfsizligi_2017 Muharrir: Tex. muharrir: Musavvir: Musahhih: Kompyuterda sahifalovchi: Sh.Aliyeva M.Holmuhamedov D.Azizov N.Hasanova N.Raxmatullayeva.

Igamberdiyev X.Z. - Ahmedova O.P. - "Unicon.UZ" DUK, Kriptografiya ilmiy-tadqiqot bolim i boshlig'i.

oddiy.blogspot.com, normatov.com 'dan foydalanildi.

G'ayratjon Rayimjonov.

Alisher Ismailov "OOP" fanidan ma'ruza matnlar to'plami.