1.2-§. Obyektga yo‘naltirilgan dasturlash tillari.

Darsning maqsadi

Ta’limiy -talabalarga obyektga yo‘naltirilgan dasturlash tillari dasturiy mahsulotlarni yaratish imkoniyatlarini tushuntirish .

Tarbiyaviy - talabalarni raqamli texnologiyaladan foydalanishdagi axboriy madaniyat qoidalari, me’yorlarini o’rgatish.

Rivojlantiruvchi – talabalarda dasturlash tillari sanoat sohasida qo‘llanish imkoniyatlarini rivojlanitirish.

Darsning turi: ma’ruza

Darsda qo’llaniladigan metodlar: Case-Study

Darsning jihozi: Kompyuter, video proyektor, slaydlar, tarqatma materiallar;

Darsdan kutiladigan natijalar:

- ✓ Obyektga yo‘naltirilgan dasturlash tillari qo’llanish sohasini o‘rganish.
- ✓ Obyektga yo‘naltirilgan dasturlashda obyekt,vorislik, poli nima.
- ✓ Yuqori darajali dasturlash tillari qo‘llanilishi**.**
- ✓ Interpretatorlar va kompilyatorlari**.**

Baholash metodlari: o‘quvchilar bilimini baholash DTS larida belgilangan nizomga muvofiq baholanadi.

Dasrga ajratilgan vaqt : 80 minut

1.2-§. Obyektga yo‘naltirilgan dasturlash tillari.

Reja:

1. Obyektga yo‘naltirilgan dasturlash tillari qo’llanish sohasini o‘rganish.
2. Obyektga yo‘naltirilgan dasturlashda obyekt,vorislik, poli nima.
3. Yuqori darajali dasturlash tillari qo‘llanilishi.
4. Interpretatorlar va kompilyatorlari.

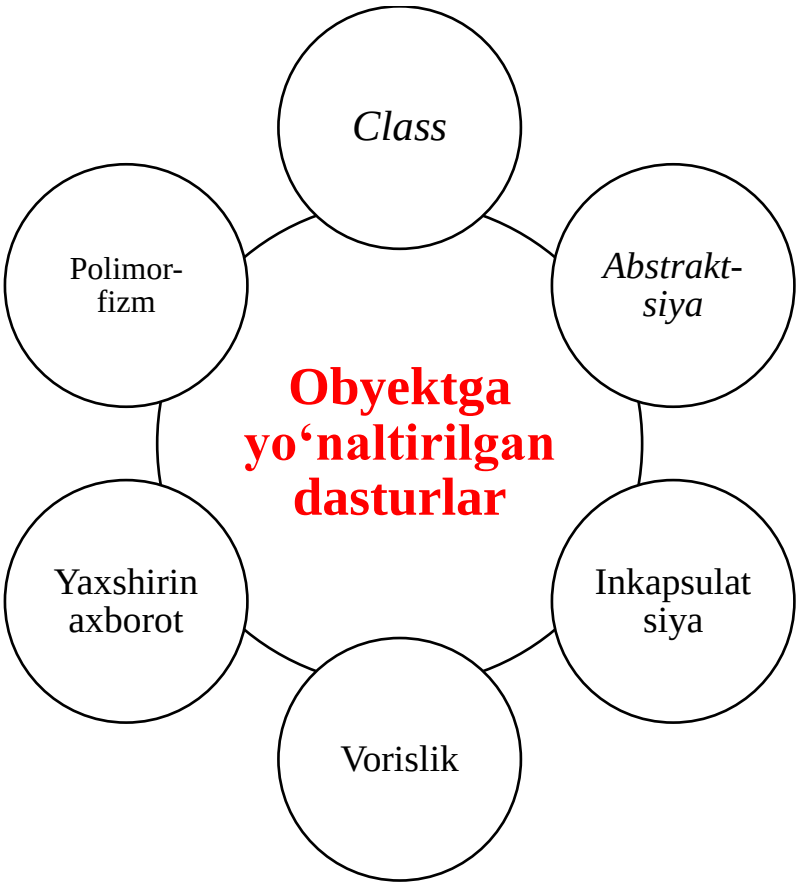
Obyektga yo‘naltirilgan dasturlash tillari. Obyektga yo‘naltirilgan dasturlar amaliy masalalarni yechishga hamda obyekt va jarayonlarni boshqarish uchun dasturiy mahsulotlarni ishlab chiqarishga mo‘ljallangan. Obyektga yo‘naltirilgan dasturlarning protsedurali dasturlash tillari (masalan, Pascal, Basic, Fortran) dan asosiy farqi shundaki, til asosan obyektlarga asoslangan holda ishlasa, protsedurali dasturlash tillari esa funksiyalarga asoslanadi. Bunda har bir buyruqlar qadamma-qadam bajarilib boriladi.

Obyektga yo‘naltirilgan dasturlarda, loyihalar o‘zaro bog‘langan obyektlar asosida yaratiladi.

Obyekt – bu obyektga yo‘naltirilgan dasturlash texnologiyasining asosiy tushunchalaridan biri hisoblanadi. Bunda, haqiqiy hayotdagi quyidagi obyektlarni misol sifatida ko‘rsatish mumkin: uy, stol, mashina, televizor va h.k.

Ularning barchasi xususiyatlari va bajaradigan vazifalari (funksiyalari) bilan farqlanadi. Masalan, mashina, xususiyatlari: tezligi, rangi, nomi, narxi; funksiyalari: yurishi, to‘xtashi, oyna tozalagichlarining ishlashi, eshiklarni ochilib yopilishi va h.k. Bu kabi hayotiy misollarning xususiyatlari va funksiyalarini aniqlash obyektga yo‘naltirilgan dasturlash nuqtai nazaridan fikrlashda muhim ahamiyat kasb etadi.

Case-Study-1: o‘quv vaziyatlar – aniq vaqt mobaynidagi vaziyatni tavsiflash, muammoni aniqlash va aniq ifodalashni shakllantirish. Obyektga yo‘naltirilgan dasturlarda qo‘llaniladigan asosiy xususiyatlari.



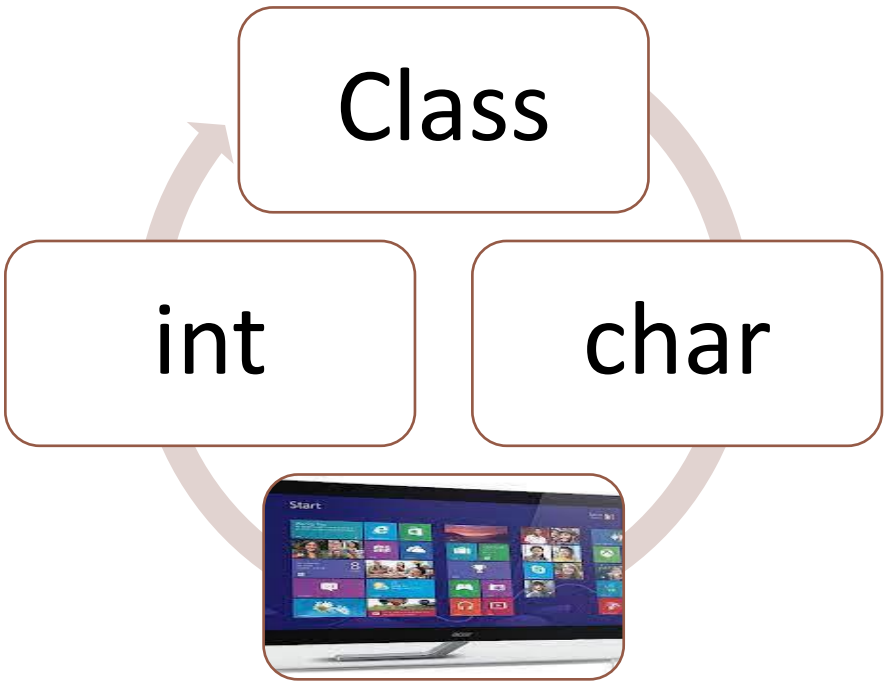
1.1-rasm. Obyektga yo‘naltirilgan dasturlashning asosiy tushunchalari.

Ushbu rasmda keltirilgan tushunchalarning asosiy xususiyatlari bilan tanishib chiqamiz.

Class – bu obyektga yo‘naltirilgan dasturlarning asosi hisoblanadi. Class har xil kodlar va ma’lumotlarni qay tarzda o‘zgarishini ifodalovchi xususiyatlardan iborat Aniqroq qilib aytadigan bo‘lsak, hayotiy obyektlarning qanday faoliyat yuritishi, nimalardan iborat ekanligi, qanday xususiyatlarga ega ekanligini tavsiflovchi kichik bir hujjat sifatida qarash ham mumkin.

Class tarkibiga o‘zgaruvchilar va metodlar (funksiyalar) hamda qiymati o‘zgarmaydigan konstantalarni kiritish, shuningdek, har bir classdan o‘zgaruvchi tipi sifatida ham foydalanish mumkin.

Case-Study-2: *O‘quv vaziyatlar – oldingi variantdan murakkabroq vaziyat tavsiflanadigan muammo yetarlicha aniqlanmagan*. “Obyektga yo’naltirilgan dasturlar”ni **Class** o‘zgaruvchi tipi, qo‘llanilishi.



1.2-rasm

Xuddi **int**, **char** yoki boshqa tiplar kabi har bir **Class** ham ma’lum bir tip sifatida qaraladi. Biror obyektni yaratish uchun Classdan foydalanilganda kompyuter xotirasidan joy ajratiladi.

C++ tilining har qanday obyekti bir xil atributlarga ega bo‘lib, bunda class obyektlarini xususiyatlari to‘liq foydalanuvchi tomonidan ishlab chiqiladi.

Foydalanuvchi biron-bir obyektga yo‘naltirilgan dasturlash muhitida ishlash jarayonida, uning standart classlari kutubxonasi (masalan, C++Builder vizual komponentlar kutubxonasi)dan foydalanishi mumkin.

C++ tilida classdan foydalanish uchun, ma’lumotlar ustida ish olib boradigan hamda obyektlarning xususiyatlarini belgilaydigan tarkibiy qismlar va metodlarni inkapsulyatsiya qilish talab qilinadi.

Case - Study3: O‘quv vaziyatlarni namoyish qilish. Masalan: “Casio” soatini obyektini tuzing.

Yechish: “Casio” soatini obyekt sifatida ko‘rib chiqamiz. Soatning suyuq kristalli displeyi, ushbu obyektning o‘zgaruvchi ma’lumoti, boshqarish tugmachalari esa obyektni funksiyasi sifatida qaraladi. Soat tugmachalarini bosib, displeyda vaqtni o‘rnatish ishlarini olib borish mumkin, ya’ni obyektga

yo‘naltirilgan dasturlash atamalarini qo‘llaydigan bo‘lsak, metodlar, ma’lumotlarni o‘zgartirib, obyekt xolatini modifikatsiya qilanadi.

Obyekt class bilan farqli tushuncha hisoblanadi. Obyekt foydalanuvchi yozgan classdagi har xil qoidalarga bo‘ysunadigan ma’lumot bo‘lib, u tezkor xotirada saqlanadi. Class esa qattiq diskda saqlanadi. Har bir loyihalashtirilgan obyekt tezkor xotiraning ma’lum bir yacheykalariga joylashadi.

Obyekt tushunchasi bilan bir vaqtda C++ Builderda komponentlar tushunchasi ham kiritilgan. Komponentlar - maxsus **class**lar bo‘lib, ularning xususiyatlari obyektlar atributlarini tashkil qiladi, metodlari esa komponentli **class**larning tegishli nusxalari ustidagi amallarni bajaradi. Metod tushunchasi odatda komponentli **class**lar tarkibida qo‘llanadi va oddiy classning funksiya o‘zgaruvchisidan farq qilmaydi. C++Builder muhiti komponentlarning turi va funksional imkoniyatlarini nafaqat metodlar yordamida, balki komponentlar **class**lariga xos bo‘lgan xususiyatlarga oid amallar bajarish imkonini ham beradi. Shuning uchun C++ Builder muhitida foydalanuvchi ilovani loyihalash va bajarish bosqichida ham komponentli obyekt ustida amallar bajarishi mumkin.

Komponentlar xususiyatlari – bu obyekt parametrlarini o‘zgartiradigan qiymatlar to‘plamidir.

Obyektga yo‘naltirilgan dasturlash tillarida, jumladan C++ Builder muhiti xususiyatlarni e’lon qilishda maxsus xizmatchi so‘zlardan foydalanadi. C++ Buider muhitida komponenta xususiyatlarini o‘zgartirish hodisalar yordamida amalga oshiriladi. C++ Builder muhitida ishlab chiqilayotgan dasturlardagi metodlar asosan hodisalarni qayta ishlashda qo‘llanadi. Bunda komponentalar o‘z xususiyatlari, metodlari va voqealarini inkapsuyatsiya qila oladi.

Komponentalar C++ tilining boshqa obyektli classlaridan, bir qator xususiyatlarni hisobga olmaganda, farq qilmaydi. Bu xususiyatlarning ba’zi birlari bilan tanishib chiqamiz: ko‘plab komponentalar foydalanuvchi interfeysni boshqarish elementi bo‘lib, ularning ba’zilari murakkab xususiyatga ega; barcha komponentalar asosiy **class** (TComponent) ning bevosita yoki bilvosita tarkibiy elementlari hisoblanadi; komponentalar odatda bevosita qo‘llanadi, ya’ni, ularning

xususiyatlari ustida ish olib boriladi; komponentalar faqat **new** operatori yordamida dinamik xotirasida joylashtiriladi; kopmonentalar xususiyatlari dinamik turlar identifikatsiyasini o‘z ichiga oladi; komponentalarni komponentalar palitrasiga qo‘shish va bundan so‘ng C++ Builder muhitida turli loyihalar yaratishda foydalanish mumkin.

Ushbu keltirilgan xususiyatlarni o‘zgartirishda metodlardan foydalaniladi. Metod - class tarkibiga kiritilgan oddiy funksiya hisoblanadi. Metodga murojaat qilish uchun, ushbu class tarkibida yoki ishlov beriladigan loyihaning dastur kodida funksiya(metod) nomini yozish talab etiladi.

Aynan metodning class bilan o‘zaro bog‘liqliligi, uni oddiy funksiya tushunchasidan ajratib turadi. Metodni bajarish paytida, u o‘z classining barcha ma’lumotlariga kirish huquqiga ega bo‘ladi.

Dasturlash amaliyotida class bilan birgalikda vorislik tushunchasi ham qo‘llaniladi.

Vorislik – bu mavjud bo‘lgan classning butun yoki qisman funksionalligi asosida yangi class yaratishdir.

Vorislik g‘oyasi obyektlarning xususiyatlarini modifikatsiyalash muammosiga yechim bo‘lib, obyektga yo‘naltirilgan dasturlashga qo‘shimcha imkoniyatlar va moslashuvchanlikni ta’minlaydi. Vorislik deyarli hech qanday cheklanishlarsiz biror foydalanuvchi tomonidan yaratilgan classlarni kengaytirish imkonini beradi. Eng oddiy classlardan boshlab, murakkablik jihatidan ortib boradigan, ammo sozlanishi ham oson, ichki tuzilishi ham oddiy bo‘lgan classlarni yaratish mumkin.

Ayniqsa yirik dasturiy loyihalarni ishlab chiqishda vorislik tamoyilini hayotga tatbiq etish soddalashib boruvchi tuzilmaviy dasturlash (umumiydan juz’iyga) texnikasi bilan moslashadi. Bunda dastur kodining murakkabligi sezilarli darajada kamayadi. Shuni yodda tutish kerakki, vorislik asosida bazaviy classdan yangi xosila class ishlab chiqiladi.

Xosila class o‘z bazaviy classining hamda classlar tabaqalanishidagi, dastlabki class xususiyatlarini, metodlari va voqealarini voris qilib oladi. Xosila class – bu

bazaviy class asosida yaratilgan, imkoniyatlari nisbatan ko‘p bo‘lgan yangi class hisoblanadi.

Vorislik paytida bazaviy class yangi atributlar va amallar hisobiga yanada rivojlanadi. Xosila classda odatda yangi ma’lumotlar, xususiyatlar va metodlar paydo bo‘ladi. Obyektlar bilan ishlashda dasturchi aniq masalani hal etish uchun mos keladigan classni tanlaydi hamda undan bir yoki bir nechta voris classni yaratadi. Funksiyalar esa xosila classga, barcha tashqi classlar ma’lumotlariga kirish huquqini olish imkonini beradi.

Bundan tashqari, voris qilib olinayotgan metodlarda, ularning bazaviy classdagi vazifasi xosila classga to‘g‘ri kelmasa, xosila class ortiqcha yuklanishi mumkin.

Odatda obyektga yo‘naltirilgan dasturlashda ortiqcha yuklanishdan foydalaniladi. Agar metod bittadan ortiq bir nomdagi funksiya bilan bog‘lansa, u ortiqcha yuklangan deb hisoblanadi.

Case – Study 3: O‘quv vaziyatlarni namoyish qilish. Masalan: “Casio” soatini obyektini tuzing, degan muammoni qo‘yilgan edi. Mavzu davomida quyidagicha yechimi ko‘rishimiz mumkin.

Yechimi: Vorislik kontseptsiyasini yuqorida keltirilgan soat haqidagi misolga tatbiq qilamiz. Vorislik tamoyiliga amal qilgan “Casio” firmasi soatning yangi modelini ishlab chiqishga qaror qildi. Bu modelning tugmachalaridan biri ikki marta bosilsa, vaqtni ovozda ayta oladi. Gapiradigan soatlar modeli (obyektga yo‘naltirilgan dasturlash atamalari bo‘yicha, yangi class)ni yangidan yaratish o‘rniga muhandislar ishni, uning nusxasidan boshlaydilar (obyektga yo‘naltirilgan dasturlash atamalari bo‘yicha, bazaviy classning yangi nusxasini yaratadi). Xosila obyekt asosiy obyektning barcha atributlari va funksionalligini voris sifatida qabul qiladi. Ovoz bilan aytilgan sonlar yangi obyektning ma’lumotlari hisoblanadi. Tugmachalarning obyektli metodlari esa ularning qo‘shimcha funksionalligini ishga tushirish uchun, ortiqcha yuklatilgan bo‘lishi kerak. Tugmachalarning ikki marta bosilish hodisasiga yangi metod javob berib, u joriy vaqtga mos keladigan sonlar

ketma-ketligi (yangi ma’lumotlar a’zolari) ning talaffuz qilinishida namoyon bo‘ladi.

Obyektga yo‘naltirilgan dasturlashda vorislikdan tashqari inkapsulyatsiya tushunchasi ham mavjud.

Inkapsulyatsiya – bu metodlar va ma’lumotlarni bir butun qilib bog‘lashdir.

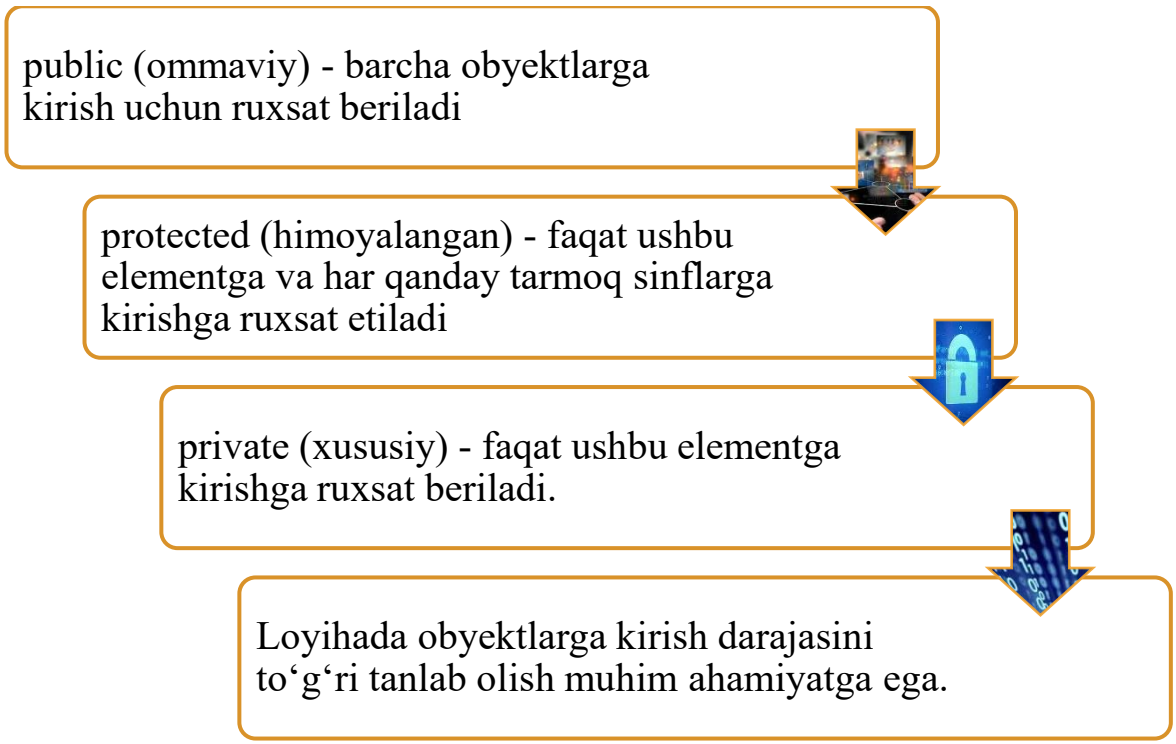
Inkapsulyatsiyalash – bu ma’lumotlarni qayta ishlaydigan dastur kodlarini bitta obyektda birlashtirishdir. Obyektga yo‘naltirilgan dasturlashda ma’lumotlar, obyekt ma’lumotlari tarkibi hisoblanadi. Kodlar esa obyektli metodlar yoki funksiya tarkibi deyiladi.

Inkapsulyatsiyalash obyektni tashqi muhitdan maksimal darajada ajratish imkonini beradi. Bu ishlab chiqilayotgan dasturlarning ishonchliligini sezilarli darajada oshiradi, chunki obyektda mujassamlangan (lokallashtirilgan) funksiyalar dastur kodiga nisbatan kam hajmdagi ma’lumotlarni almashinadi. Bunda ushbu ma’lumotlarning miqdori va turi nazorat qilinadi.

Inkapsulyatsiyalashning muhim jihatlaridan biri, obyektlar almashinuvi va ularning bir dasturdan ikkinchisiga o‘tkazilishi soddaligi bilan farqlanadi. Obyektga yo‘naltirilgan dasturlashda inkapsulyatsiya tamoyilining soddaligi va qulayligi dasturchilarni obyektga yo‘naltirilgan dasturlash muhitlari, xususan C++ Builder dasturlash muhiti tarkibiga kiruvchi vizual komponentlar kutubxonasini kengaytirish imkoniyatini yaratadi.

Har bir elementni ommaviy interfeysga kiritish yoki, aksincha, undan chiqarish uchun maxsus so‘zlardan foydalanish talab etiladi. Obyektga yo‘naltirilgan dasturlashning har bir tilida maxsus so‘zlar to‘plami belgilangan bo‘lib, bu so‘zlar asosan bir xil funksiyalarni bajaradi.

Case-Stady: *O‘quv vaziyatlarni namoyish qilish.* Obyektga mo‘ljallangan tillarning aksariyatida kirish darajasi:



1.3-rasm: Obyektga mo‘ljallangan tillarning aksariyatida kirish darajasi.

Obyektga yo‘naltirilgan dasturlashda vorislik va inkapsulyatsiya bilan birgalikda polimorfizm tushunchasi ham qo‘llaniladi.

Polimorfizm – bu bir interfeysning turli xil ko‘rinishda yaratalishidir.

Inkapsulatsiyalash va vorislikni obyektga yo‘naltirilgan dasturlashning foydali vositalari sifatida olib qarash mumkin bo‘lsa, polimorfizm esa eng universal va harakatchan vositadir. Polimorfizm inkapsulatsiyalash va vorislik bilan chambarchas bog‘liq bo‘lib, polimorfizmsiz obyektga yo‘naltirilgan dasturlash samarali bo‘lmaydi. Polimorfizm – obyektga yo‘naltirilgan dasturlash paradigmasida markaziy tushunchadir. Obyektga yo‘naltirilgan dasturlashda murakkab loyihalar yaratish uchun polimorfizm tushunchasi bilan batafsil tanishish maqsadga muvofiq hisoblanadi.

Polimorfizm shunday holatki, bunda qandaydir bir element ko‘p shakllarga ega bo‘lishi mumkin. Dasturlash tilida “ko‘p shakllar” deyilganda, bir element turli kodlarning nomidan ish ko‘rishi tushuniladi. Shu bois, polimorfizm yordamida bitta nom turli xususiyatni bildirishi mumkin.

Polimorfizm yordamida tizimga ixtiyoriy vaqtda qo‘shimcha funksiyalarni qo‘shish mumkin. Bunda polimorfizmning quyidagi uchta asosiy turi qo‘llaniladi: qo‘shilish polimorfizmi; parametrik polimorfizm; ortiqcha yuklanish.

Qo‘shilish polimorfizmidan foydalanib, yangi tarmoq sinflarni kiritgan holda, tizimning xususiyatini o‘zgartirish mumkin. Uning bosh afzalligi shundaki, dastlabki dasturni o‘zgartirmay turib, yangi xususiyat yaratiladi.

Parametrik polimorfizmdan foydalanib turdosh metodlar va universal turlar yaratish mumkin. Turdosh metodlar va turlar ma’lumotlarning ko‘plab toifalari asosida dasturiy mahsulotlarni yaratish imkonini beradi. Parametrik polimorfizm yordamida parametr turini e’lon qilmay turib turdosh metodlar yaratish mumkin. Metodlarning parametrik xususiyatlari bo‘lgani kabi, turlarning o‘zi ham parametrik xususiyatga ega bo‘lishi mumkin. Biroq polimorfizmning bunday turi barcha tillarda ham uchrayvermaydi, ammo bu C++ tilida mavjud.

Ortiqcha yuklanish turida bitta nom turlicha metodlarni bildirishi mumkin. Bunda metodlar faqat miqdorlar va parametr turlari bilan farqlanadi. Metod o‘z argumentlariga bog‘liq bo‘lmaganda, ortiqcha yuklanish samarali hisoblanadi. Metod o‘ziga xos parametr turlari bilan cheklanmaydi, balki har xil turdagi parametrlarga nisbatan ham qo‘llaniladi. Masalan, **max** metodini ko‘rib chiqaylik. **max** ikkita muayyan parametrlarni qabul qilib, ularning qaysi biri katta ekanini aniqlaydi. Bunda metod parametri butun yoki haqiqiy sonlarni qabul qiladi.

Polimorfizmdan foydalanganda dasturni bajarish vaqtida tizim tomonidan tekshiruvlar o‘tkaziladi. Bu tekshiruvlar turlari statik ravishda berilgan qiymatlarga ishlov berishga qaraganda ko‘proq vaqt talab qiladi. Polimorfizm tushunchasi bilan birgalikda obyektga yo‘naltirilgan dasturlashda abstraktsiya tushunchasi ham qo‘llaniladi.

Abstraktsiya obyektga yo'naltirilgan dasturlash tillarining asosiy tushunchalaridan biridir. Uning asosiy maqsadi foydalanuvchidan keraksiz ma'lumotlarni yashirish orqali murakkablikni soddalashtirishdan iborat. Bu foydalanuvchiga barcha yashirin murakkablikni tushunmasdan, undan foydalanishga imkon yaratadi.

Dasturlashdagi obyekt. Dasturlashdagi obyekt ham haqiqiy hayotdagi obyektlarga o‘xshash. Ular ham qandaydir xususiyatlar va bajaradigan funksiyalardan iboratdir. Obyektning xususiyatlari har xil dasturiy o‘zgaruvchilardan iborat bo‘lib, ularni o‘zgartirish uchun qandaydir funksiyalar bajariladi. Bunda obyektlarning asosiy elementlaridan biri sifatida metod tushunchasi qo‘llaniladi.

Metodlar (funksiyalar) obyektning imkoniyatlarini izohlaydi, masalan, yuqorida avtomobil haqida misol keltirilgan. Unda avtomobilni yurishi va boshqa harakatlarini keltirganmiz. Ushbu harakatlarni metodlar orqali ifodalaymiz, masalan, quyida yur() metodi keltirilgan:

```
public class Avtomashina {
public void yur() {
  }
}
```

Obyektga yo‘naltirilgan dasturlashda metod bilan birgalikda o‘zgaruvchi tushunchasi ham qo‘llaniladi.

Obyektlarni o‘zgaruvchi sifatida qarab, uning qiymatlari o‘zgartirish mumkin. Masalan, **class**da mashinaning yurish tezligi o‘zgaradi. Bunda, tezlig qandaydir sonlarda o‘zgaradi. Masalan yur() metodidan to‘xta() metodiga o‘tganda, uning tezligi o‘zgaradi. Bunga o‘xshash obyektlar o‘zgaruvchilar deb yuritiladi.

Qiymatlari o‘zgarmaydigan obyektlar konstantalar deb ataladi.

Konstantalar bir marta qiymat berilib, qayta o‘zgarmaydigan obyektlar hisoblanadi. Masalan quyida berilgan **Feruz** nomli classda, uning tug‘ilgan yili o‘zgarmaydi. Ushbu yaratiladigan classning dastur kodi quyidagicha yoziladi:

```
public class Feruz{
int tezlik=0;
final int tugilgan_yili=1990;
// final kalit so‘zi obyektning konstanta ekanligini bildiradi
public void yur(){
```

}

}

Keltirilgan classdan foydalanilanib yaratilgan dasturda ***tugilgan_yili*** ga yangi qiymat berilsa, dastur xatolik haqida xabar beradi. Chunki dastlab final xizmatchi so‘zi orqali e’lon qilingan.

Dasturlarning obyektga yo‘naltirilgan paradigmasi. Dasturlash tillari yaratilgandan buyon soha olimlari tomonidan bir necha dasturlash tillari yaratilib kelinmoqda. Vaqt o‘tib, dasturlash tillarining yanada rivojlangan turlari paydo bo‘lganligi sababli, dastur kodlarini yozish uchun ishlatiladigan usul ham o‘zgarib bormoqda. Metodologiya, algoritmlar, kontsepsiyalar yillar mobaynida rivojlanishi tufayli, dasturlash sohasiga paradigma tushunchasi kiritildi.

Paradigma (yunonchadan olingan "namuna, model") - ilmiy jamoatchilik tomonidan qabul qilingan hamda birgalikda foydalaniladigan, fundamental ilmiy qarashlar, tushunchalar va atamalar to‘plamidir.

Dasturlash paradigmasi – bu kompyuter dasturlarini yozish uslubini belgilaydigan g‘oyalar va tushunchalar to‘plamidir. Bu hisoblashni tashkil qilish va kompyuter bajaradigan vazifalarni belgilaydigan kontseptsiya usulidir.

Zamonaviy dasturlash paradigmalarini to‘rt qismga ajratish mumkin: protsessual dasturlash; obyektga yo‘naltirilgan dasturlash; komponentga yo‘naltirilgan dasturlash; aspektga yo‘naltirilgan dasturlash.

Obyektga yo‘naltirilgan dasturlash ko‘p funksiyaligi bilan boshqa dasturlash tillaridan afzalroq hisoblanadi. Hozirgi kunda barcha dasturlash tillari sanoatning asosini tashkil etadi.

Savollar va mustaqil bajarish uchun topshiriqlar

1. Obyektga yo‘naltirilgan dasturlash tillari haqida ma’lumot bering.

2. Obyektga yo‘naltirilgan dasturlashning asosiy tushunchalari haqida ma’lumot bering.

3. Obyektga yo‘naltirilgan dasturlashda obyekt nima?

3. **class** haqida ma’lumot bering.

4. Vorislik tushunchasini izohlang.

5. Inkapsulyatsiya haqida ma’lumot bering.
6. Polimorfizm haqida ma’lumot bering.
7. Obyektga yo‘naltirilgan dasturlashda abstraktsiya nima?
8. Obyektga yo‘naltirilgan dasturlashda komponentlar nima vazifa bajaradi?
9. Komponentlarning asosiy xususiyatlarini sanang.
10. Obyektga yo‘naltirilgan dasturlashda vorislik deganda nimani tushunasiz?