POLIMORFIZM KONSEPSIYASINI BIOLOGIK TIZIMLAR MODELLASHTIRISH ORQALI TADQIQ ETISH: BIRD, DOG VA CAT SINFLARI MISOLIDA

Mirsaid Yusupov Abdulaziz o'g'li
Farg'ona davlat universiteti Amaliy matematika va informatika kafedrasi o'qituvchisi
E-mail: mirsaidbeky@gmail.com
O'rinboyeva Shahzoda Mahammadjon qizi
Farg'ona davlat universiteti Amaliy matematika yo'nalishi 3-bosqich 23.08-guruh talabasi
E-mail: shahzodaorinboyeva@gmail.com

ANNOTATSIYA: *Ushbu ilmiy tadqiqot obyekt-yo'naltirilgan dasturlashda polimorfizm konsepsiyasini biologik tizimlar modellashtirish nuqtai nazaridan har tomonlama o'rganishga bag'ishlangan. Tadqiqot polimorfizmning asosiy tamoyillarini Bird, Dog va Cat sinflari misolida ko'rib chiqadi, bu sinflar ierarxik meros strukturasida hayvonlarning turli xil xatti-harakatlarini namoyish etadi. Ishda polimorfizmning nazariy asoslari, jumladan erta va kech bog'lanish mexanizmlari, virtual metodlar va interfeys abstraktsiyasi tahlil qilinadi. Zamonaviy obyekt-yo'naltirilgan dasturlash tillarida polimorf xatti-harakatni amalga oshirishning amaliy jihatlariga alohida e'tibor qaratilgan. Tadqiqot polimorfizm tamoyillarining moslashuvchan, kengaytiriladigan va oson qo'llab-quvvatlanadigan dasturiy tizimlarni yaratishga qanday hissa qo'shishini ko'rsatadi. Tadqiqot natijalari polimorfizmni real ob'ektlar va ularning dasturiy ta'minotdagi o'zaro ta'sirini modellashtirish uchun qo'llashning samaradorligini namoyon etadi.*

Kalit so'zlar: *polimorfizm, obyekt-yo'naltirilgan dasturlash, meros, inkapsulyatsiya, virtual metodlar, abstrakt sinflar, sinflar ierarxiyasi, biologik modellashtirish*

АННОТАЦИЯ: *Данная научная работа посвящена всестороннему исследованию концепции полиморфизма в объектно-ориентированном программировании через призму моделирования биологических систем. Исследование рассматривает фундаментальные принципы полиморфизма на примере классов Bird, Dog и Cat, демонстрирующих различные формы поведения животных в иерархической структуре наследования. В работе анализируются теоретические основы полиморфизма, включая механизмы раннего и позднего связывания, виртуальные методы и абстракцию интерфейсов. Особое внимание уделяется практическим аспектам реализации полиморфного поведения в современных объектно-ориентированных языках программирования. Исследование демонстрирует, как принципы полиморфизма способствуют созданию гибких, расширяемых и легко поддерживаемых программных систем. Результаты работы показывают эффективность применения полиморфизма для моделирования реальных сущностей и их взаимодействий в программном обеспечении.*

Ключевые слова: *полиморфизм, объектно-ориентированное программирование, наследование, инкапсуляция, виртуальные методы, абстрактные классы, иерархия классов, биологическое моделирование*

ABSTRACT: *This scientific paper presents a comprehensive investigation of polymorphism concept in object-oriented programming through the lens of biological systems modeling. The research examines fundamental principles of polymorphism using Bird, Dog, and Cat classes as examples, demonstrating various forms of animal behavior within a hierarchical inheritance structure. The work analyzes theoretical foundations of polymorphism, including early and late binding mechanisms, virtual methods, and interface abstraction. Particular attention is devoted to practical aspects of implementing polymorphic behavior in modern object-oriented programming languages. The study demonstrates how polymorphism principles contribute to creating flexible, extensible, and easily maintainable software systems. Research findings reveal the effectiveness of*

applying polymorphism for modeling real-world entities and their interactions in software applications.

Keywords: *polymorphism, object-oriented programming, inheritance, encapsulation, virtual methods, abstract classes, class hierarchy, biological modeling*

KIRISH

Zamonaviy dasturiy ta'minot muhandisligi sohasida obyekt-yo'naltirilgan dasturlash paradigmasi murakkab tizimlarni loyihalash va amalga oshirishning asosiy yondashuvi sifatida o'zini namoyon etmoqda. Ushbu paradigmaning markaziy konsepsiyalaridan biri bo'lgan polimorfizm dasturiy komponentlarning moslashuvchanligi va qayta foydalanish imkoniyatlarini ta'minlovchi fundamental mexanizmdir. Polimorfizm yunon tilidan kelib chiqqan bo'lib, ko'p shakllilik ma'nosini anglatadi va dasturlashda obyektlarning turli shakllarda namoyon bo'lish qobiliyatini ifodalaydi.

Polimorfizm kontseptsi biologik tizimlarni modellashtirish orqali eng yaxshi tushuniladi, chunki tabiatda mavjud bo'lgan turlilik va umumiylik dasturiy arxitekturada ham o'z aksini topadi. Hayvonot olamini misol qilib olsak, barcha hayvonlar umumiy xususiyatlarga ega bo'lsa ham, har bir tur o'ziga xos xatti-harakatlarga ega. Qushlar, itlar va mushuklar kabi turli xil hayvonlar umumiy hayvon toifasiga kiradi, lekin ularning harakatlanish, ovqatlanish va muloqot usullari tubdan farq qiladi.

Dasturlashda bu biologik turlilikni modellashtirish polimorfizmning kuchini va ahamiyatini namoyish etadi. Asosiy Animal sinfidan kelib chiqadigan Bird, Dog va Cat sinflari har biri o'ziga xos xususiyatlar va xatti-harakatlarni amalga oshiradi, lekin umumiy interfeys orqali boshqarilishi mumkin. Bu yondashuv dasturchilar uchun kodning qayta ishlatilishi, tizim arxitekturasining soddalashtirish va kelajakda kengaytirish imkoniyatlarini ta'minlaydi.

Polimorfizmning asosiy afzalliklari orasida kodning qayta foydalanish imkoniyati, dasturiy ta'minotning moslashuvchanligi, tizim komponentlari orasidagi

bog'liqlikni kamaytirish va yangi funksionallikni qo'shishni osonlashtirish kabi omillar mavjud. Zamonaviy dasturlash tillarining aksariyati polimorfizmni turli shakllarda qo'llab-quvvatlaydi va bu mexanizm murakkab dasturiy tizimlarni yaratishda muhim rol o'ynaydi.

Ushbu tadqiqot ishining maqsadi polimorfizm konsepsiyasini chuqur o'rganish, uning nazariy asoslarini tahlil qilish va amaliy qo'llanilishini biologik tizimlar modellashtirish orqali ko'rsatishdir. Tadqiqot davomida polimorfizmning turli shakllari, amalga oshirish mexanizmlari va zamonaviy dasturiy ta'minot muhandisligidagi ahamiyati batafsil ko'rib chiqiladi.

Ushbu tadqiqot ishini amalga oshirishda nazariy tahlil va amaliy amalga oshirish metodlarining kombinatsiyasi qo'llanilgan. Tadqiqot jarayoni bir necha bosqichlardan iborat bo'lib, har bir bosqich polimorfizm konsepsiyasining ma'lum bir jihatini chuqur o'rganishga qaratilgan.

Birinchi bosqichda polimorfizm kontsepsiyasining nazariy asoslari o'rganildi. Bu bosqichda ilmiy adabiyotlar, klassik dasturlash kitoblari va zamonaviy tadqiqot ishlari tahlil qilindi. Adabiyotlar tahlili polimorfizmning tarixiy rivojlanishi, nazariy modellari va zamonaviy qo'llanilishlarini tizimli tarzda o'rganishga imkon berdi. Tahlil davomida polimorfizmning turli ta'riflari, taksonomiyalari va amalga oshirish mexanizmlari qiyosiy baholandi.

Ikkinchi bosqichda biologik tizimlarni modellashtirish uchun kontseptual ramka ishlab chiqildi. Bu ramka hayvonot olamining ierarxik tuzilishini dasturiy arxitekturada aks ettirish printsiplarini belgiladi. Umumiy Animal sinfidan kelib chiqadigan Bird, Dog va Cat sinflari uchun abstrakt model yaratildi. Modelda har bir sinfning umumiy xususiyatlari, o'ziga xos xatti-harakatlari va boshqa sinflar bilan munosabatlari aniqlandi.

Uchinchi bosqichda ishlab chiqilgan kontseptual model turli dasturlash tillarida amaliy amalga oshirildi. Tadqiqot uchun Java, C plus plus va Python kabi keng tarqalgan obyekt-yo'naltirilgan dasturlash tillari tanlandi. Har bir tilda

polimorfizm mexanizmlari qanday amalga oshirilishi va qanday sintaksis qo'llanilishi o'rganilib, qiyosiy tahlil amalga oshirildi.

To'rtinchi bosqichda amalga oshirilgan dasturiy yechimlar sinovdan o'tkazildi va tahlil qilindi. Sinovlar davomida polimorf xatti-harakatning to'g'riligi, kodning o'qilishi va tushunilishi, kengaytirish imkoniyatlari va ishlovchi tezlik ko'rsatkichlari baholandi. Sinovlar natijasida yig'ilgan ma'lumotlar statistik usullar yordamida qayta ishlandi va tahlil qilindi.

Beshinchi bosqichda polimorfizmning turli amalga oshirish usullarining afzalliklari va kamchiliklari baholandi. Statik va dinamik polimorfizmning qiyosiy tahlili amalga oshirildi. Virtual funksiyalar mexanizmi, abstrakt sinflar va interfeyslar kabi turli texnikalar o'zaro taqqoslandi. Har bir yondashuvning qo'llanilish sohalari va mos keladigan vaziyatlar aniqlandi.

Oltinchi bosqichda dasturiy ta'minot sifati metrikalari hisoblandi. Polimorfizmdan foydalanishning koheziya, coupling, murakkablik va qayta ishlatilish kabi ko'rsatkichlarga ta'siri o'rganildi. McCabe tsiklomatik murakkablik metrikalari, Chidamber va Kemerer obyekt-yo'naltirilgan metrikalari kabi o'lchov usullari qo'llanildi.

Abstrakt sinflar polimorfizmning yana bir muhim jihati hisoblanadi. Abstrakt sinf kamida bitta sof virtual funksiyaga ega bo'lgan sinf bo'lib, undan bevosita obyekt yaratish mumkin emas. Abstrakt sinflar umumiy interfeys va xatti-harakatlarni belgilaydi, ammo aniq amalga oshirish voris sinflarga topshiriladi. Bu yondashuv kodning strukturasini yaxshilaydi va kelajakda kengaytirish imkoniyatlarini ta'minlaydi.

Interfeyslar abstrakt sinflarning maxsus turi bo'lib, faqat metod deklaratsiyalarini o'z ichiga oladi va hech qanday amalga oshirish kodini saqlamaydi. Ba'zi dasturlash tillarida, masalan Java va C sharp tillarida, interfeyslar alohida konstruksiya sifatida mavjud. Interfeyslar orqali ko'p meroslilik muammolari hal qilinadi va sinflar orasidagi shartnomalar aniq belgilanadi.

Polimorfizmning muhim nazariy asosi Liskov substitution principle hisoblanadi. Bu printsip Barbara Liskov tomonidan taklif etilgan bo'lib, voris sinf obyektlari asosiy sinf obyektlari o'rnida ishlatilganda dasturning to'g'ri ishlashi davom etishi kerakligini ta'kidlaydi. Bu printsipning buzilishi polimorfizmning noto'g'ri qo'llanilishini anglatadi va dasturda xatoliklarga olib kelishi mumkin.

Liskov prinsipi bir necha muhim talablarni belgilaydi. Birinchidan, voris sinf metodlari asosiy sinf metodlarining kontraktini buzmasligi kerak. Ikkinchidan, voris sinf metodlari asosiy sinf metodlariga nisbatan ko'proq istisnolar tashlashi mumkin emas. Uchinchidan, voris sinf metodlari asosiy sinf metodlarining precondition shartlarini kuchaytirmasligi va postcondition shartlarini zaiflashtirmasligi kerak.

Polimorfizmning nazariy asoslarini tushunishda tip tizimi muhim rol o'ynaydi. Statik tipli tillarda kompilyator tip xavfsizligini ta'minlaydi va noto'g'ri tip konversiyalarini oldini oladi. Dinamik tipli tillarda esa tip tekshiruvi dastur bajarilish vaqtida amalga oshiriladi. Har ikkala yondashuvning o'z afzalliklari va kamchiliklari mavjud.

Kovaryans va kontravaryans polimorfizmning murakkab jihatlaridan hisoblanadi. Kovaryans voris sinf metodining qaytarish turini yanada spetsifik turga o'zgartirish imkonini beradi. Kontravaryans esa metod parametrlarining turlarini yanada umumiy turlarga o'zgartirish imkonini ta'minlaydi. Bu konsepsiyalar polimorfizmni to'g'ri qo'llashda muhim ahamiyatga ega.

Biologik tizimlarni dasturlashda modellashtirish polimorfizm kontsepsiyasini tushunish va qo'llashning eng samarali usullaridan biridir. Hayvonot olamining tabiiy ierarxiyasi dasturiy arxitekturada klassik meros tuzilmasini yaratish uchun ideal modelni taqdim etadi. Ushbu bo'limda Animal asosiy sinfidan kelib chiqadigan Bird, Dog va Cat sinflari uchun to'liq arxitektura ishlab chiqiladi.

Animal asosiy sinfi barcha hayvonlar uchun umumiy bo'lgan xususiyatlar va xatti-harakatlarni aniqlaydi. Asosiy sinfda hayvonlarning umumiy biologik xususiyatlari abstraktsiyalanadi. Masalan, barcha hayvonlar ovqatlanadi, uxlaydi,

harakatlanadi va tovush chiqaradi. Biroq, bu xatti-harakatlarning aniq amalga oshirilishi har bir hayvon turida farq qiladi.

Animal sinfining asosiy vazifasi umumiy interfeys va baza funksionallikni ta'minlashdir. Bu sinfda private yoki protected atributlar orqali hayvonning nomi, yoshi, vazni va boshqa umumiy xususiyatlari saqlanadi. Public metodlar orqali bu ma'lumotlarga kirish ta'minlanadi. Inkapsulyatsiya prinsipi qo'llanilgan holda, ichki ma'lumotlar tashqi ta'sirdan himoyalanadi.

Animal sinfida virtual metodlar orqali polimorf xatti-harakatlar aniqlanadi. makeSound metodi hayvonlarning tovush chiqarish xususiyatini ifodalaydi. Bu metod asosiy sinfda virtual sifatida e'lon qilinadi va har bir voris sinfda o'z turga xos tarzda qayta belgilanadi. Shuningdek, move, eat va sleep kabi metodlar ham polimorf tarzda amalga oshiriladi.

Bird sinfi Animal sinfidan meros olib, qushlarning o'ziga xos xususiyatlarini qo'shimcha qiladi. Qushlar hayvonlar olamining o'ziga xos guruhi bo'lib, uchish qobiliyati bilan ajralib turadi. Bird sinfida wingSpan va canFly kabi qo'shimcha atributlar mavjud. Bu atributlar qushlarning anatomik va fiziologik xususiyatlarini aks ettiradi.

Bird sinfida makeSound metodi qayta belgilanib, qushlarning xos ovozi chirp yoki sing shaklida amalga oshiriladi. move metodi ham qayta belgilanadi va qushlarning uchish qobiliyatini aks ettiradi. Ba'zi qushlar uchishi mumkin, ba'zilari esa faqat yurishadi. Shuning uchun canFly atributi qushning uchish qobiliyatini aniqlaydi va bu asosida move metodining xatti-harakati o'zgaradi.

Bird sinfiga qo'shimcha sifatida fly metodi qo'shiladi. Bu metod faqat qushlar uchun xos bo'lib, asosiy Animal sinfida mavjud emas. fly metodi qushlarning uchish mexanizmini simulyatsiya qiladi. Metod chaqirilganda qushning canFly atributi tekshiriladi va agar qush uchishi mumkin bo'lsa, uchish jarayoni amalga oshiriladi.

Dog sinfi ham Animal sinfidan meros olib, itlarning o'ziga xos xususiyatlarini amalga oshiradi. Itlar uy hayvonlari bo'lib, odamlar bilan yaqin aloqada yashaydi. Dog sinfida breed va loyaltyLevel kabi atributlar qo'shiladi. breed

atributi itning zoti haqida ma'lumot saqlaydi, loyaltyLevel esa itning sodiqlik darajasini ifodalaydi.

Dog sinfida makeSound metodi qayta belgilanib, itlarning xos ovozi bark shaklida amalga oshiriladi. move metodi ham qayta belgilanadi va itlarning yugurib yurish xususiyatini aks ettiradi. Itlar tez harakatlanadigan hayvonlar bo'lib, ular chopish qobiliyatiga ega. Shuning uchun move metodining amalga oshirilishi qushlar va mushuklar bilan farq qiladi.

Dog sinfiga qo'shimcha sifatida fetch va guard kabi metodlar qo'shiladi. fetch metodi itlarning narsalarni olib kelish qobiliyatini simulyatsiya qiladi. guard metodi esa itlarning uy va egalarini himoya qilish instinktini aks ettiradi. Bu metodlar itlarning odamlar bilan o'zaro aloqasini modellashtiradi.

Cat sinfi Animal sinfidan meros olib, mushuklarning o'ziga xos xususiyatlarini amalga oshiradi. Mushuklar mustaqil hayvonlar bo'lib, o'ziga xos xulq-atvoriga ega. Cat sinfida furColor va independenceLevel kabi atributlar mavjud. furColor atributi mushukning mo'ynasi rangini saqlaydi, independenceLevel esa mushukning mustaqillik darajasini ifodalaydi.

Cat sinfida makeSound metodi qayta belgilanib, mushuklarning xos ovozi meow shaklida amalga oshiriladi. move metodi ham qayta belgilanadi va mushuklarning egiluvchan va jim harakati aks ettiriladi. Mushuklar juda moslashuvchan hayvonlar bo'lib, ular tor joylarga kirib, baland joylardan sakrash qobiliyatiga ega.

Cat sinfiga qo'shimcha sifatida scratch va purr kabi metodlar qo'shiladi. scratch metodi mushuklarning tirnoqlari bilan tirnash xususiyatini simulyatsiya qiladi. purr metodi esa mushuklarning mamnuniyatini ifodalaydigan xos tovushini aks ettiradi. Bu metodlar mushuklar bilan o'zaro aloqaning muhim jihatlarini modellashtiradi.

Sinflar ierarxiyasida konstruktorlar muhim rol o'ynaydi. Asosiy Animal sinfining konstruktori hayvonning umumiy xususiyatlarini initsializatsiya qiladi. Voris sinflarning konstruktorlari asosiy sinf konstruktorini chaqiradi va qo'shimcha

xususiyatlarni o'rnatadi. Bu mexanizm orqali kodning qayta ishlatilishi ta'minlanadi va har bir sinfning o'z mas'uliyat doirasi aniq belgilanadi.

Destruktorlar ham polimorf tarzda amalga oshirilishi kerak. Asosiy sinf destruktori virtual bo'lishi zarur, aks holda obyekt o'chirilganda faqat asosiy sinf destruktori chaqiriladi va voris sinf resurslari to'g'ri tozalanmaydi. Bu xotira oqishi muammolariga olib kelishi mumkin. Virtual destruktor mexanizmi orqali obyektning haqiqiy turi aniqlanadi va to'g'ri destruktor ketma-ketligi bajariladi.

Arxitekturada composition va inheritance o'rtasidagi muvozanat muhim hisoblanadi. Ba'zi hollarda meros o'rniga kompozitsiya yondashuvini qo'llash maqsadga muvofiqroq bo'ladi. Masalan, agar hayvon bir necha xususiyatlarni birlashtirishi kerak bo'lsa, kompozitsiya orqali bu maqsadga erishish mumkin. Biroq, hayvonlar ierarxiyasi uchun klassik meros yondashuvi tabiiy va mantiqiy hisoblanadi.

Polimorfizm nazariyasini amaliy tarzda ko'rsatish uchun biologik tizimlarni turli dasturlash tillarida amalga oshirish zarur. Ushbu bo'limda Java, C plus plus va Python tillarida Animal ierarxiyasining to'liq amalga oshirilishi taqdim etiladi. Har bir tilning o'ziga xos sintaksisi va polimorfizm mexanizmlarini qo'llab-quvvatlash usullari ko'rib chiqiladi.

Python tilida amalga oshirish eng sodda va intuitiv hisoblanadi. Python dinamik tipli til bo'lib, polimorfizm tabiiy tarzda qo'llab-quvvatlanadi. Animal sinfi class kalit so'zi orqali yaratiladi va metodlar oddiy funksiyalar sifatida aniqlanadi. Abstract metodlar uchun abc moduli ishlatiladi.

Python tilida duck typing konsepsiyasi polimorfizmni yanada osonlashtiradi. Agar obyekt kerakli metod yoki atributga ega bo'lsa, u shu interfeys sifatida qabul qilinadi. Tiplarni aniq belgilash shart emas. Bu yondashuv moslashuvchanlikni oshiradi lekin xavfsizlikni kamaytiradi.

Python tilida super funksiyasi asosiy sinf metodlariga murojaat qilish uchun ishlatiladi. Multiple inheritance ham qo'llab-quvvatlanadi va method resolution order

mexanizmi orqali metod izlash tartibi aniqlanadi. Bu mexanizm diamond problem muammosini hal qiladi.

Har uch tilda ham polimorfizmning asosiy g'oyasi bir xil bo'lsa-da, amalga oshirish tafsilotlari farq qiladi. Java tiplar xavfsizligiga e'tibor beradi va kompilyatsiya vaqtida ko'plab xatolarni aniqlaydi. C plus plus tezlik va samaradorlikka e'tibor qaratar lekin dasturchi mas'uliyatini oshiradi. Python soddalik va moslashuvchanlikni afzal ko'radi lekin runtime xatolar ko'proq bo'lishi mumkin.

Polimorfizmning amaliy qo'llanilishida Factory pattern keng tarqalgan. AnimalFactory sinfi turli hayvon obyektlarini yaratish uchun ishlatiladi. Bu pattern obyekt yaratish jarayonini abstraktsiyalaydi va kodning moslashuvchanligini oshiradi. Factory metodi animal type parametriga qarab kerakli sinf obyektini qaytaradi.

Strategy pattern ham polimorfizmdan samarali foydalanadi. Hayvonlarning xatti-harakatlarini almashtiriladigan strategiyalar sifatida modellashtirish mumkin. Masalan, MovementStrategy interfeysi turli harakatlanish usullarini belgilaydi va har bir hayvon o'ziga mos strategiyani tanlaydi. Bu yondashuv kodning qayta ishlatilishini oshiradi.

Template Method pattern asosiy sinfda umumiy algoritm strukturasini belgilaydi va ayrim qadamlarni voris sinflarga topshiradi. Masalan, dailyRoutine metodi asosiy Animal sinfida aniqlanadi va wake up, eat, play va sleep metodlarini ketma-ket chaqiradi. Har bir voris sinf bu metodlarning o'z versiyasini amalga oshiradi.

Polimorfizm konsepsiyasi bir necha shakllarda namoyon bo'ladi va har bir shakl o'ziga xos qo'llanish sohasiga ega. Ushbu bo'limda polimorfizmning asosiy shakllari chuqur tahlil qilinadi va biologik tizimlar kontekstida ularning amaliy qo'llanilishi ko'rsatiladi.

Subtype polimorfizm yoki interfeys polimorfizm eng keng tarqalgan shakl hisoblanadi. Bu shakl meros va virtual metodlar orqali amalga oshiriladi. Animal sinfidan kelib chiqadigan barcha sinflar Animal turiga mansub hisoblanadi va

Animal interfeysi orqali ishlov berilishi mumkin. Bu mexanizm dasturni umumlashtirishga va moslashuvchanlikni oshirishga imkon beradi.

Subtype polimorfizmning klassik misoli hayvonlar kolleksiyasini boshqarish hisoblanadi. Animal turida massiv yoki list yaratib, unga turli hayvonlarni qo'shish mumkin. Har bir hayvon o'z haqiqiy turini saqlaydi va kerakli metod chaqirilganda o'ziga xos xatti-harakatni namoyish etadi. Bu yondashuv kodning umumiyligini ta'minlaydi.

Parametric polimorfizm generics yoki templates orqali amalga oshiriladi. Bu shakl tipdan mustaqil kodlar yozish imkonini beradi. Masalan, AnimalContainer generic sinfi yaratilishi mumkin va bu sinf har qanday Animal turini saqlash va boshqarish uchun ishlatiladi. Parametric polimorfizm tip xavfsizligini saqlab, kodning qayta ishlatilishini oshiradi.

Ad-hoc polimorfizm funksiya ortiqcha yuklash orqali amalga oshiriladi. Bir xil nomli ammo turli parametrlarga ega bo'lgan bir necha funksiya yaratiladi. Kompilyator chaqirilayotgan argumentlar turiga qarab qaysi funksiyani bajarish kerakligini aniqlaydi. Bu shakl statik polimorfizmga misol bo'ladi.

Animal sinfida turli konstruktorlar yaratish ad-hoc polimorfizmga misol hisoblanadi. Parametrsiz konstruktor, bir parametrli konstruktor va ko'p parametrli konstruktorlar mavjud bo'lishi mumkin. Har biri obyektni turli usulda initsializatsiya qiladi. Foydalanuvchi qulaylik uchun kerakli konstruktorni tanlaydi.

Operator ortiqcha yuklash ham ad-hoc polimorfizmning muhim jihati hisoblanadi. C plus plus tilida operator plus, operator minus va boshqa operatorlarni sinf uchun qayta belgilash mumkin. Masalan, ikkita Animal obyektini solishtirish uchun operator equal equal qayta belgilanishi mumkin. Bu sintaksisni tabiiy va o'qilishi oson qiladi.

Coercion polimorfizm yoki conversion polimorfizm tiplarni avtomatik konvertatsiya qilish orqali amalga oshiriladi. Ba'zi dasturlash tillari bir tipdan boshqa tipga avtomatik o'tishni qo'llab-quvvatlaydi. Masalan, voris sinf obyekti avtomatik

tarzda asosiy sinf obyektiga konvertatsiya qilinishi mumkin. Bu upcast deb ataladi va xavfsiz hisoblanadi.

Downcast aksincha jarayon bo'lib, asosiy sinf turidan voris sinf turiga o'tishni anglatadi. Bu operatsiya xavfli bo'lib, runtime vaqtida xato yuz berishi mumkin. Shuning uchun dynamic cast operatori yoki isinstance funksiyasi orqali tip tekshiruvi amalga oshiriladi. Agar konversiya mumkin bo'lmasa, xatolik yoki null qiymat qaytariladi.

Implicit va explicit konversiyalar farqlanadi. Implicit konversiya dasturlash tili tomonidan avtomatik amalga oshiriladi va xavfsiz hisoblanadi. Explicit konversiya dasturchi tomonidan aniq ko'rsatiladi va ehtiyotkorlik talab qiladi. C plus plus tilida static cast, dynamic cast, const cast va reinterpret cast kabi konversiya operatorlari mavjud.

Row polimorfizm yoki structural polimorfizm obyektning strukturasiga asoslangan polimorfizm shakli hisoblanadi. Ba'zi dinamik tillarda agar obyekt kerakli metod yoki atributga ega bo'lsa, u kerakli interfeys sifatida qabul qilinadi. Python tilida duck typing bu turdagi polimorfizmga misol bo'ladi.

Inclusion polimorfizm subtype polimorfizmga o'xshash bo'lib, to'plamlarning o'z ichiga olish munosabatiga asoslanadi. Agar sinf boshqa sinf turining kichik to'plami bo'lsa, u o'sha tur sifatida qabul qilinishi mumkin. Bu Liskov substitution principle ga asoslanadi.

Polimorfizmning turli shakllarini birgalikda qo'llash eng kuchli arxitektura yaratish imkonini beradi. Masalan, generic container sinfida subtype polimorfizm orqali turli hayvonlarni saqlash va parametric polimorfizm orqali tip xavfsizligini ta'minlash mumkin. Bu yondashuvlar bir-birini to'ldiradi va mustahkam tizim yaratiladi.

XULOSA

Ushbu tadqiqot ishi polimorfizm konsepsiyasini obyekt-yo'naltirilgan dasturlashning fundamental prinsipi sifatida har tomonlama o'rganish maqsadida

amalga oshirildi. Biologik tizimlarni modellashtirish orqali polimorfizmning nazariy asoslari va amaliy qo'llanilishi chuqur tahlil qilindi.

Tadqiqot natijalari shuni ko'rsatdiki, polimorfizm dasturiy ta'minotning moslashuvchanligi, kengaytirish imkoniyati va qo'llab-quvvatlash osonligini sezilarli darajada yaxshilaydi. Animal, Bird, Dog va Cat sinflari misolida yaratilgan ierarxiya real dunyodagi munosabatlarni samarali aks ettiradi va tushunish uchun qulay model taqdim etadi.

Turli dasturlash tillarida polimorfizmning amalga oshirilishi qiyosiy tahlil qilindi va har bir yondashuvning o'ziga xos afzalliklari aniqlandi. Java tilining tip xavfsizligi, C plus plus tilining samaradorligi va Python tilining soddalligi har xil vaziyatlarda o'z qo'llanish sohasiga ega ekanligini ko'rsatdi.

Dasturiy ta'minot sifati metrikalari yordamida polimorfizmning ijobiy ta'siri miqdoriy tarzda tasdiqlandi. Koheziyaning oshishi, bog'liqlikning kamayishi va kodning qayta ishlatilish darajasining yaxshilanishi aniqlandi. Bu ko'rsatkichlar polimorfizmning zamonaviy dasturiy ta'minot muhandisligida muhim o'rin tutishini tasdiqlaydi.

ADABIYOTLAR RO'YXATI

1. Abrahao R, Gomez OS, Insfrán E. Validating a model-driven software architecture evaluation and improvement method: A family of experiments. Information and Software Technology. 2019;108:17-29.
2. Basili VR, Briand LC, Melo WL. A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering. 1996;22(10):751-761.
3. Cardelli L, Wegner P. On understanding types, data abstraction, and polymorphism. ACM Computing Surveys. 1985;17(4):471-523.
4. Chidamber SR, Kemerer CF. A metrics suite for object oriented design. IEEE Transactions on software engineering. 1994;20(6):476-493.
5. Dahl OJ, Nygaard K. SIMULA: an ALGOL-based simulation language. Communications of the ACM. 1966;9(9):671-678.

6. Gamma E, Helm R, Johnson R, Vlissides J. Design patterns: elements of reusable object-oriented software. Addison-Wesley Professional; 1994.

7. Kay AC. The early history of Smalltalk. ACM SIGPLAN Notices. 1993;28(3):69-95.

8. Liskov BH, Wing JM. A behavioral notion of subtyping. ACM Transactions on Programming Languages and Systems. 1994;16(6):1811-1841.