OBYEKTGA YO‘NALTIRILGAN DASTURLASHDA __INIT__ METODINING FUNKSIONAL ROLI VA AMALIY AHAMIYATI

Xurshida Nabiyeva Iqboljon qizi
Farg‘ona davlat universiteti talabasi
xurshidaanabiyeva@gmail.com
Mirsaid Yusupov Abdulaziz o‘g‘li
Farg‘ona davlat universiteti amaliy matematika va informatika kafedrasi o‘qituvchisi
mirsaidbeky@gmail.com

ANNOTATSIYA

Ushbu maqolada obyektga yo‘naltirilgan dasturlashda __init__ metodining ahamiyati nazariy va amaliy jihatdan tahlil qilinadi. Obyekt yaratish jarayonida yuzaga keladigan muammolar ochib berilib, ularni konstruktor mexanizmi orqali samarali hal etish yo‘llari asoslab beriladi. Tahlil natijalari __init__ metodidan to‘g‘ri foydalanish obyektlarning mantiqiy yaxlitligini ta’minlash, dastur barqarorligini oshirish va kod sifatini yaxshilashda muhim rol o‘ynashini ko‘rsatadi.

KALIT SO‘ZLAR: obyektga yo‘naltirilgan dasturlash, klass va obyekt, __init__ metodi, konstruktor, inkapsulyatsiya, obyekt hayot sikli, Python dasturlash tili

АННОТАЦИЯ

В данной статье анализируется важность метода __init__ в объектно-ориентированном программировании с теоретической и практической точек зрения. Раскрываются проблемы, возникающие в процессе создания объектов, и обосновываются способы их эффективного решения с использованием механизма конструктора. Результаты анализа показывают, что правильное использование метода __init__ играет важную роль в обеспечении логической целостности объектов, повышении стабильности программы и улучшении качества кода.

КЛЮЧЕВЫЕ СЛОВА: объектно-ориентированное программирование, класс и объект, метод __init__, конструктор, инкапсуляция, жизненный цикл объекта, язык программирования Python

ANNOTATION

This article analyzes the importance of the __init__ method in object-oriented programming from both a theoretical and practical perspective. The problems that arise during the process of object creation are revealed and ways to effectively solve them using the constructor mechanism are justified. The results of the analysis show that the correct use of the __init__ method plays an important role in ensuring the logical integrity of objects, increasing program stability and improving code quality.

KEYWORDS: object-oriented programming, class and object, __init__ method, constructor, encapsulation, object life cycle, Python programming language

KIRISH

Zamonaviy dasturiy tizimlar tobora murakkablashib borayotgan bir sharoitda obyektga yo‘naltirilgan dasturlash dasturchilar uchun muammolarni mantiqiy va tartibli tarzda hal qilish imkonini beruvchi asosiy yondashuvlardan biri sifatida qaraladi. Biroq amaliyotda ko‘plab dasturchilar obyektlarni yaratish jarayonida ularning ichki holatini to‘g‘ri shakllantirish masalasiga yetarlicha e’tibor bermaydi. Natijada obyektlar aniq boshlang‘ich qiymatlarsiz yaratiladi, atributlar keyinchalik tartibsiz tarzda o‘rnatiladi va bu holat dastur ishonchliligi hamda tushunarliligiga salbiy ta’sir ko‘rsatadi. Aynan shu nuqtada obyektning dastlabki holatini belgilovchi mexanizmning ahamiyati dolzarb muammo sifatida namoyon bo‘ladi.

Obyektga yo‘naltirilgan yondashuvda har bir obyekt ma’lum bir holat va xatti-harakatlar majmuasini ifodalaydi. Ushbu holat obyekt yaratilgan zahoti aniq va mantiqan to‘g‘ri bo‘lishi talab etiladi, aks holda keyingi jarayonlarda kutilmagan xatoliklar yuzaga kelishi mumkin. Amaliy tajriba shuni ko‘rsatadiki, ko‘plab xatolar obyekt yaratilish bosqichidayoq noto‘g‘ri qarorlar qabul qilinganidan kelib chiqadi. Bu esa boshlang‘ich sozlash jarayonining nafaqat texnik, balki konseptual jihatdan ham muhim ekanini ko‘rsatadi.

Python dasturlash tilida obyektning dastlabki holatini belgilash vazifasi __init__ metodi orqali amalga oshiriladi. Ushbu metod ko‘pincha faqat atributlarga qiymat beruvchi oddiy funksiya sifatida qabul qilinadi, ammo uning roli bundan ancha kengroq. To‘g‘ri tashkil etilgan __init__ metodi obyektning mantiqiy yaxlitligini ta’minlaydi, kodni o‘qish va kengaytirishni yengillashtiradi hamda dasturiy arxitekturaning barqarorligini mustahkamlaydi. Shu bois konstruktor mexanizmini chuqurroq tahlil qilish va undan samarali foydalanish yo‘llarini ko‘rsatish muhim ahamiyat kasb etadi.

Mazkur yo‘nalishda asosiy e’tibor obyektlar yaratish jarayonida yuzaga keladigan muammolarni aniqlash va ularni __init__ metodidan oqilona foydalanish orqali bartaraf etish imkoniyatlarini ochib berishga qaratiladi. Kirish qismida yoritilgan ushbu muammo keyingi bo‘limlarda nazariy asoslar va amaliy misollar orqali izchil ravishda tahlil qilinadi hamda samarali yechimlar bilan boyitiladi.

Obyektga yo‘naltirilgan dasturlashning asosiy g‘oyasi real dunyodagi tushunchalarni dasturiy obyektlar orqali modellashtirishdan iborat bo‘lib, bu jarayonda obyektning ichki holati muhim o‘rin tutadi. Har bir obyekt ma’lum atributlarga ega bo‘ladi va ushbu atributlar obyekt yaratilgan zahotiyoq mantiqan to‘g‘ri qiymatlarga ega bo‘lishi lozim. Agar obyekt boshlang‘ich holatsiz yoki noaniq qiymatlar bilan yaratilsa, dastur keyingi bosqichlarda kutilmagan xatti-harakatlarni namoyon qilishi

mumkin. Shu sababli obyektni yaratish jarayoni faqat xotirada joy ajratish bilan cheklanmay, balki uning ichki holatini izchil va boshqariladigan tarzda shakllantirishni ham o‘z ichiga oladi.

Python dasturlash tilida obyekt yaratish mexanizmi klass asosida amalga oshiriladi va bu jarayon avtomatik ravishda __init__ metodi bilan bog‘liq holda ishlaydi. Ushbu metod obyekt yaratilgan paytda chaqiriladi va atributlarga boshlang‘ich qiymatlar berish vazifasini bajaradi. Nazariy jihatdan qaralganda, __init__ obyektning hayot siklidagi eng muhim nuqtalardan birini belgilaydi, chunki aynan shu bosqichda obyekt qanday holatda ishlashi aniqlanadi. Shunga qaramay, amaliyotda bu metod ko‘pincha yetarlicha rejalashtirilmasdan yoziladi yoki umuman chetlab o‘tiladi, natijada obyektlar o‘zaro mantiqiy bog‘liqlikni yo‘qotadi.

Muammo shundan iboratki, __init__ metodiga e’tibor berilmaganda atributlar keyinchalik tashqi muhit orqali o‘zgartiriladi, bu esa inkapsulyatsiya tamoyilining buzilishiga olib keladi. Bunday yondashuv kodni tushunishni qiyinlashtiradi, obyektning haqiqiy holatini aniqlashni murakkablashtiradi va xatolarni yashirin holatda ko‘payishiga sabab bo‘ladi. Nazariy tahlil shuni ko‘rsatadiki, obyektning ichki holati boshidan aniq belgilansa, dastur barqarorligi va mantiqiy yaxlitligi sezilarli darajada oshadi.

Yuqoridagi konseptual bog‘lanishlardan kelib chiqib aytish mumkinki, __init__ metodi obyekt va klass o‘rtasidagi muhim bog‘lovchi mexanizm vazifasini bajaradi. U nafaqat texnik jihatdan obyektni ishga tayyorlaydi, balki dasturiy dizaynning mantiqiy asosini ham belgilaydi. Shu nuqtai nazardan, konstruktorni puxta rejalashtirish va undan ongli foydalanish obyektga yo‘naltirilgan dasturlashdagi asosiy yechimlardan biri sifatida qaraladi. Keyingi bo‘limda ushbu nazariy muammo qanday qilib amaliy yechimlar orqali bartaraf etilishi batafsil yoritiladi.

Nazariy tahlilda aniqlangan muammolarni bartaraf etishning eng samarali yo‘li obyekt yaratish jarayonini ongli va tizimli ravishda tashkil etishdan iborat bo‘ladi. Python dasturlash muhitida bu vazifa to‘liq __init__ metodi orqali amalga oshiriladi. Konstruktor yordamida obyektning barcha muhim atributlari bir joyda, aniq qoidalar asosida va mantiqiy bog‘liqlikda boshlang‘ich holatga keltiriladi. Bu yondashuv obyektning keyingi ishlash jarayonida barqaror va oldindan bashorat qilinadigan xatti-harakatini ta’minlaydi.

Oddiy amaliy vaziyatni ko‘rib chiqilsa, masalan, talaba obyektini modellashtirish jarayonida ism, kurs va baho kabi atributlar mavjud bo‘ladi. Agar ushbu atributlar obyekt yaratilgandan so‘ng alohida-alohida berilsa, obyekt ma’lum vaqt davomida to‘liq bo‘lmagan holatda qoladi. __init__ metodi orqali esa bu atributlar obyekt yaratilgan zahoti majburiy tarzda aniqlanadi va obyekt hech qachon noaniq holatda mavjud bo‘lmaydi. Natijada mantiqiy xatoliklar kamayadi va dastur oqimi soddalashadi.

Murakkabroq tizimlarda __init__ metodining ahamiyati yanada yaqqol namoyon bo‘ladi. Masalan, bank hisob raqami, foydalanuvchi profili yoki texnik qurilma modelida obyektning boshlang‘ich holati noto‘g‘ri belgilansa, bu butun tizim ishlashiga salbiy ta’sir ko‘rsatishi mumkin. Konstruktor orqali atributlar o‘rtasidagi bog‘liqlikni nazorat qilish, noto‘g‘ri qiymatlarni filtrlash va obyektni faqat yaroqli holatda yaratish imkoniyati yuzaga keladi. Bu holat __init__ metodini nafaqat boshlovchi vosita, balki himoya mexanizmi sifatida ham talqin qilish imkonini beradi.

Amaliy tahlil natijalari shuni ko‘rsatadiki, __init__ metodidan to‘g‘ri foydalanish dastur tuzilmasini tartibga soladi, obyektlar o‘rtasidagi mantiqiy aloqalarni aniq belgilaydi va kodni qayta ishlashni yengillashtiradi. Bunday yondashuv obyektga yo‘naltirilgan dasturlashning inkapsulyatsiya va mas’uliyatni ajratish tamoyillari bilan to‘liq uyg‘unlashadi. Shu sababli konstruktorni puxta loyihalash muammoning asosiy yechimi sifatida namoyon bo‘ladi. Keyingi bo‘limda ushbu yechimning samaradorligi va boshqa yondashuvlar bilan qiyosiy jihatlari muhokama qilinadi.

Amaliy yechimlar tahlili shuni ko‘rsatadiki, __init__ metodidan ongli va tizimli foydalanish obyektga yo‘naltirilgan dasturlashda uchraydigan ko‘plab muammolarni bartaraf etadi. Eng avvalo, obyektning ichki holati yaratilish bosqichidayoq aniq belgilanganligi sababli, dastur davomida atributlarning tasodifiy yoki mantiqsiz qiymatlar olishi ehtimoli keskin kamayadi. Bu holat dastur xatti-harakatining barqaror bo‘lishiga va kutilmagan xatoliklarning oldini olishga xizmat qiladi.

Muhokama jarayonida aniqlanishicha, konstruktor orqali boshqarilgan obyektlar bilan ishlash kodni o‘qish va tushunishni sezilarli darajada yengillashtiradi. Dasturchi obyekt qanday ma’lumotlar asosida yaratilishini aniq ko‘ra oladi va uning holati qayerda hamda qachon shakllanganini kuzatishi osonlashadi. Aksincha, atributlar tashqi muhitda yoki turli funksiyalar orqali berilgan holatlarda obyektning haqiqiy holatini aniqlash murakkablashadi va kodni qo‘llab-quvvatlash xarajati ortadi. Shu nuqtai nazardan, __init__ metodi dasturiy tozalikka xizmat qiluvchi muhim vosita sifatida baholanadi.

Shuningdek, __init__ metodidan foydalanish obyektga yo‘naltirilgan dizayn tamoyillari bilan ham bevosita bog‘liq. Inkapsulyatsiya tamoyili obyektning ichki holatini tashqi ta’sirlardan himoyalashni talab qiladi va konstruktor aynan shu vazifani bajarishda muhim rol o‘ynaydi. Obyekt faqat mantiqan yaroqli holatda yaratilgani sababli, uning keyingi ishlashi ham izchil bo‘ladi. Bu yondashuv mas’uliyatni ajratish tamoyiliga ham mos keladi, chunki obyektning boshlang‘ich holatini belgilash mas’uliyati aniq bitta nuqtada jamlanadi.

Qiyosiy tahlil shuni ko‘rsatadiki, boshlang‘ich qiymatlarni alohida sozlash metodlari orqali berish yoki obyekt yaratilgandan so‘ng atributlarni o‘zgartirish kabi yondashuvlar qisqa muddatda qulay ko‘rinsa-da, uzoq muddatli istiqbolda murakkablik va xatolarni ko‘paytiradi. Konstruktor asosidagi yondashuv esa dastur

arxitekturasini mustahkamlaydi va tizim kengaytirilganda ham o‘z samaradorligini saqlab qoladi. Shu bois __init__ metodidan foydalanish faqat texnik tanlov emas, balki strategik qaror sifatida qaralishi maqsadga muvofiqdir.

Xulosa

Obyektga yo‘naltirilgan dasturlashda obyektning boshlang‘ich holatini to‘g‘ri shakllantirish dastur barqarorligi va mantiqiy yaxlitligini ta’minlovchi asosiy omillardan biri hisoblanadi. Tahlillar shuni ko‘rsatadiki, __init__ metodi obyekt yaratish jarayonida markaziy o‘rin egallab, atributlarni aniq va nazorat qilinadigan holatda belgilash imkonini beradi. Konstruktor mexanizmidan oqilona foydalanish obyektlarning noaniq holatda mavjud bo‘lishi bilan bog‘liq muammolarni bartaraf etadi, inkapsulyatsiya tamoyilini mustahkamlaydi va kodning o‘qiluvchanligi hamda kengaytiriluvchanligini oshiradi. Natijada __init__ metodi faqat texnik vosita emas, balki obyektga yo‘naltirilgan dizaynning muhim konseptual elementi sifatida namoyon bo‘ladi.

Foydalanilgan adabiyotlar

1. Лутц М. *Изучаем Python*. — 5-е изд. — СПб.: Питер, 2019. — 832 с.
2. Саммерфилд М. *Python 3. Самое необходимое*. — М.: ДМК Пресс, 2018. — 560 с.
3. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. *Приёмы объектно-ориентированного проектирования*. — СПб.: Питер, 2016. — 368 с.
4. Гради Буч. *Объектно-ориентированный анализ и проектирование*. — М.: Вильямс, 2017. — 720 с.
5. Хорстманн К. *Объектно-ориентированное программирование*. — М.: Бином, 2015. — 416 с.
6. Sweigart A. *Automate the Boring Stuff with Python*. — San Francisco: No Starch Press, 2020. — 592 p.
7. Python Software Foundation. *The Python Language Reference*. — URL: https://docs.python.org (murojaat sanasi: 10.12.2025).