Latin American Journal of Education
www.lajoe.org

DASTURLASH TILLARINI O‘QITISHDA OB’EKTGA YO‘NALTIRILGAN DASTURLASHNING O‘RNI

https://doi.org/10.5281/zenodo.20318251

Muallif: **Azizjon Xurramov Baxodir o‘g‘li[1*].**
*e-mail: *khurramov@tsue.uz*
[1]*Toshkent davlat iqtisodiyot universiteti o‘qituvchisi, Toshkent, O‘zbekiston.*

Annotatsiya: *Ushbu tadqiqot oliy ta'lim muassasalarida dasturlash fanlarini o‘qitishda ob'ektga yo‘naltirilgan dasturlash (OOP) paradigmasining o‘rni va samaradorligini ilmiy jihatdan tahlil qiladi. Tadqiqot 180 nafar talaba ishtirokida o‘tkazilgan eksperiment, so‘rovnoma va pedagogik kuzatuv metodlari asosida amalga oshirildi. Olingan natijalar shuni ko‘rsatadiki, OOP tushunchalarini erta va tizimli o‘qitish talabalarning abstrakt fikrlash qobiliyatini, kodni qayta ishlatish (code reuse) ko‘nikmalarini va dasturiy ta'minot muhandisligi sohasidagi umumiy tayyorgarligini statistik jihatdan sezilarli darajada oshiradi. Tadqiqot shuningdek OOP ni o‘qitishning optimal ketma-ketligi va metodologik yondashuvlarini ham taklif etadi.*

Kalit so‘zlar: *ob'ektga yo‘naltirilgan dasturlash, OOP, dasturlash ta'limi, Java, Python, abstrakt fikrlash, kapsulalash, polimorfizm, meros, oliy ta'lim, pedagogik metodlar.*

KIRISH (INTRODUCTION)

1.1. Muammoning dolzarbligi: Hozirgi kunda dasturiy ta'minot sanoati jadal sur'atlar bilan rivojlanmoqda. GitHub statistikasiga ko‘ra, 2023-yilda dunyo bo‘ylab 100 milliondan ortiq dasturchi faoliyat yuritgan va bu ko‘rsatkich har yili o‘rtacha 8-10% ga oshib bormoqda. Shu bilan birga, sanoat vakillari bitiruvchilarning amaliy ko‘nikmalari, xususan, katta hajmli dasturiy loyihalarni loyihalash va boshqarish qobiliyati yetarli emasligidan shikoyat qilmoqdalar.

Dasturlash paradigmalari orasida ob'ektga yo‘naltirilgan dasturlash (OOP) alohida o‘rin tutadi. Tiobe Index 2024 ma'lumotlariga ko‘ra, dunyo bo‘ylab eng ko‘p ishlatiladigan dasturlash tillarining aksariyati — Java, Python, C++, C#, Ruby — OOP paradigmasiga asoslangan yoki uni qo‘llab-quvvatlaydi. Bu esa OOP ni o‘qitishni zamonaviy IT ta'limining muhim tarkibiy qismiga aylantiradi.

Biroq OOP ni o‘qitish jarayonida bir qator muammolar mavjud. Ko‘plab tadqiqotlar (Kölling, 2008; Bennedsen & Caspersen, 2007; Robins et al., 2003) dasturlashni o‘rganish — ayniqsa OOP kabi abstrakt paradigmalarni — talabalar uchun juda qiyin ekanligini isbotlagan. Talabalar ko‘pincha sinf (class) va ob'ekt (object) farqini tushunmaydi, meros (inheritance) tushunchasi ularni

Latin American Journal of Education
www.lajoe.org

chalkashtirib yuboradi, polimorfizm esa ko‘pchilik uchun tushunib bo‘lmas abstraktsiya bo‘lib qoladi.

O‘zbekiston sharoitida bu muammo yanada keskinroq. Oliy ta'lim muassasalarida OOP ni o‘qitish uchun yagona metodologik standart mavjud emas, o‘qituvchilar o‘rtasida OOP ni qachon va qanday o‘qitish kerakligi bo‘yicha konsensus yo‘q, darsliklar esa ko‘pincha eskirgan misollarga asoslanadi.

1.2. Tadqiqotning maqsadi va vazifalari:

Maqsad: Oliy ta'lim muassasalarida dasturlash tillarini o‘qitishda OOP paradigmasining o‘rni va samaradorligini empirik jihatdan aniqlash hamda samarali o‘qitish modelini ishlab chiqish.

Vazifalar:

1. OOP ni o‘qitishning mavjud metodologik yondashuvlarini tahlil qilish
2. OOP tushunchalarining o‘zlashtirilish darajasiga ta'sir etuvchi omillarni aniqlash
3. Turli o‘qitish metodlarining samaradorligini eksperiment yo‘li bilan solishtirish
4. OOP ni o‘qitishning optimal modeli va ketma-ketligini taklif etish

1.3. Tadqiqotning ilmiy yangiligi: Ushbu tadqiqot O‘zbekiston oliy ta'lim kontekstida OOP o‘qitishning empirik jihatdan sinovdan o‘tkazilgan modelini birinchi marta taklif etadi. Shuningdek, OOP tushunchalarini o‘qitishning kognitiv yuklanma (cognitive load) nazariyasi asosida optimal ketma-ketligi asoslab beriladi.

1.4. Adabiyotlar sharhi: OOP ni o‘qitish bo‘yicha xalqaro miqyosda ko‘plab tadqiqotlar olib borilgan. Kölling (2008) BlueJ muhitini ishlab chiqib, vizual ob'ekt modeli orqali OOP tushunchalarini osonlashtirish mumkinligini isbotladi. Bennedsen va Caspersen (2007) 63 universitetda o‘tkazgan keng ko‘lamli tadqiqotida dasturlash kurslarida muvaffaqiyatsizlik darajasi o‘rtacha 32% ni tashkil etishini aniqladi — bu jiddiy muammodir.

Wing (2006) "hisoblash tafakkuri" (computational thinking) tushunchasini kiritib, dasturlashni o‘rganishni shunchaki sintaksis o‘rganish emas, balki muammolarni tizimli hal qilish qobiliyatini rivojlantirish sifatida qarash kerakligini ta'kidladi. OOP bu borada ayniqsa mos keladi, chunki real dunyo muammolarini modellashtirish uchun ideal vosita bo‘lib xizmat qiladi.

Sweller (1988) tomonidan ishlab chiqilgan kognitiv yuklanma nazariyasi (Cognitive Load Theory — CLT) OOP o‘qitishda muhim ahamiyat kasb etadi. CLT ga ko‘ra, inson ish xotirasi (working memory) cheklangan hajmga ega va bir vaqtning o‘zida ko‘p yangi tushunchalarni o‘zlashtirish samaradorlikni keskin pasaytiradi. Shu sababli OOP tushunchalarini bosqichma-bosqich, to‘g'ri ketma-ketlikda o‘qitish zarur.

METODLAR (Methods)

2.1. Tadqiqot dizayni: Ushbu tadqiqot aralash metod (mixed-methods) dizaynida o‘tkazildi: miqdoriy ma'lumotlar (test natijalari, statistik

Latin American Journal of Education
www.lajoe.org

ko‘rsatkichlar) va sifatiy ma'lumotlar (kuzatuv, intervyu, so‘rovnoma) birgalikda ishlatildi. Tadqiqot 2023-yil sentabridan 2024-yil maygacha, to‘liq bir o‘quv yili davomida olib borildi.

2.2. Ishtirokchilar: Tadqiqotda Toshkent davlat iqtisodiyot universiteti (TDIU) va Toshkent amaliy fanlar universiteti (TAFU) ning dasturlash fanlarini o‘rganayotgan 1-kurs talabalari ishtirok etdi. Jami 180 nafar talaba uchta guruhga bo‘lindi:

1-guruh — Nazorat guruhi (60 kishi): An'anaviy yondashuv — avval protsedurali dasturlash (C tili), so‘ngra OOP (Java). OOP mavzulari ketma-ket: sinf → ob'ekt → meros → polimorfizm → interfeyslar.

2-guruh — A-eksperimental guruh (60 kishi): "Objects First" yondashuvi — dasturlash o‘qitishni boshidanoq OOP asosida boshlash (BlueJ muhitida Java). Meros va polimorfizm keyinroq o‘rgatiladi.

3-guruh — B-eksperimental guruh (60 kishi): Gibrid yondashuv — Python tili bilan boshlash (sodda sintaksis), OOP tushunchalarini real loyihalar orqali o‘rgatish, har bir tushuncha aniq muammo yechimiga bog'lanadi.

Guruhlar tarkibi tanlashda muvozanat ta'minlandi: har bir guruhda taxminan teng nisbatda erkak/ayol, yuqori/o‘rta/past akademik tayyorgarlikdagi talabalar bo‘ldi.

2.3. O‘quv materiali va dastur: Barcha uchala guruh uchun qamrab olinadigan mavzular bir xil bo‘ldi: OOP ning to‘rtta asosiy ustuni (kapsulalash, meros, polimorfizm, abstraktsiya), dizayn naqshlari (design patterns) asoslari, UML diagrammalari va kichik hajmli loyiha ishlab chiqish. Farq faqat o‘qitish ketma-ketligi, tillar va metodlarda edi.

Har bir guruh haftasiga 4 soat nazariy va 4 soat amaliy mashg'ulot o‘tadi. Semestr davomida 3 ta oraliq nazorat, 1 ta kurs loyihasi va yakuniy imtihon bo‘ldi.

2.4. O‘lchov va baholash vositalari:

Bilimlarni o‘lchash:

- OOP tushunchalarini tushunish testi (40 savol, validatsiya o‘tgan, Cronbach α = 0.87)
- Kurs loyihasi sifatini baholash rubrikalari (kod sifati, OOP qo‘llanilishi, hujjatlashtirish)
- Algoritmik masalalarni yechish testi (LeetCode uslubida 15 ta masala)

Motivatsiya va munosabatni o‘lchash:

- MSLQ (Motivated Strategies for Learning Questionnaire) so‘rovnomasi
- Darsga qatnashish va faollik kuzatuvi
- Semestr oxirida yarim tuzilgan intervyu (har guruhdan 10 nafar talaba)

Ma'lumot tahlili: IBM SPSS 27 dasturida bir tomonlama dispersiya tahlili (One-way ANOVA), post-hoc Tukey testi va Pearson korrelyatsiya koeffitsientlari hisoblandi. Sifatiy ma'lumotlar tematik tahlil orqali qayta ishlandi.

2.5. Etik masalalar: Barcha ishtirokchilar ixtiyoriy asosda tadqiqotga

Latin American Journal of Education
www.lajoe.org

jalb etildi. Ma'lumotlar anonim shaklda to‘plandi va faqat ilmiy maqsadlarda ishlatildi. Tadqiqot universitetning ilmiy-etika kengashi tomonidan tasdiqlandi.

NATIJALAR (Results)

3.1. OOP tushunchalarini o‘zlashtirish darajasi: Semestr yakunidagi OOP bilim testida uchala guruh o‘rtasida statistik jihatdan sezilarli farq aniqlandi ($F(2,177) = 18.43$, $p < 0.001$).

Guruh	O‘rtacha ball (100 dan)	Standart og'ish	Min	Max
Nazorat (an'anaviy)	61.4	14.2	28	89
A-eksperimental (Objects First)	74.8	11.7	42	97
B-eksperimental (Gibrid/Python)	78.3	10.1	51	99

Post-hoc Tukey testi shuni ko‘rsatdiki, nazorat guruhi bilan har ikkala eksperimental guruh o‘rtasidagi farq statistik jihatdan muhim ($p < 0.001$), ammo ikki eksperimental guruh o‘rtasidagi farq chegaraviy ($p = 0.048$).

3.2. OOP ning to‘rtta ustunini o‘zlashtirish tahlili: Har bir OOP tushunchasi bo‘yicha alohida tahlil o‘tkazildi:

OOP tushunchasi	Nazorat guruhi	A-eksperimental	B-eksperimental
Kapsulalash	71.2%	83.4%	86.1%
Meros	58.3%	72.6%	76.8%
Polimorfizm	49.7%	66.3%	71.4%
bstraktsiya	44.1%	61.8%	68.9%

Eng katta farq polimorfizm va abstraktsiya mavzularida kuzatildi — bu tushunchalar an'anaviy o‘qitishda eng qiyin o‘zlashtirilishini tasdiqlaydi. B-eksperimental guruhda real loyihalar orqali o‘rgatilgan abstraktsiya tushunchasi nazorat guruhiga nisbatan 24.8 foizga yuqori o‘zlashtirildi.

3.3. Kurs loyiha sifati: Kurs loyihalarini baholashda ekspert komissiyasi (3 ta sanoat mutaxassisi) ishtirok etdi. Loyihalar OOP qo‘llanilishi, kodni qayta foydalanish, hujjatlashtirish va umumiy arxitektura bo‘yicha 5 ballik tizimda baholandi:

Mezon	Nazorat	A-eksperimental	B-eksperimental
OOP to‘g'ri qo‘llanilishi	2.8	3.9	4.2
Kodni qayta foydalanish	2.4	3.7	4.0
Hujjatlashtirish	2.9	3.4	3.8
Umumiy arxitektura	2.6	3.8	4.1

Latin American Journal of Education
www.lajoe.org

O‘rtacha	2.7	3.7	4.0

3.4. OOP tushunchalarini o‘rganishdagi kognitiv to‘siqlar: Intervyu va kuzatuv ma'lumotlari asosida OOP o‘rganishda eng ko‘p uchraydigan kognitiv to‘siqlar aniqlandi:

Sinf va ob'ekt farqi — barcha guruhlar uchun dastlabki 3 hafta davomida eng ko‘p uchraydigan chalkashlik. Nazorat guruhida bu muammoni yengish o‘rtacha 4.2 hafta, A-eksperimental guruhda 2.8 hafta, B-eksperimental guruhda 2.1 hafta vaqt oldi.

"this" kalit so‘zi — talabalarning 67% uchun dastlab tushunarsiz. BlueJ visual muhiti (A-guruh) bu muammoni vizual ravishda ko‘rsatib hal qilishda samarali bo‘ldi.

Meros zanjiri — ko‘p bosqichli meros (A → B → C) ko‘pchilikni chalkashtirib yubordi. B-guruhda real dunyo analogiyalari (masalan: Hayvon → Sut emizuvchi → It) orqali bu muammo 40% kamroq uchraydi.

Polimorfizm va upcasting — eng qiyin tushuncha. Uchala guruhda ham qiyinchilik kuzatildi, lekin B-guruhda amaliy misol (turli shakldagi to‘lovlar tizimi loyihasi) orqali tushunchani mustahkamlash ancha samarali bo‘ldi.

3.5. Motivatsiya ko‘rsatkichlari: MSLQ so‘rovnomasi natijalari bo‘yicha o‘quv motivatsiyasi dinamikasi:

Ko‘rsatkich	Nazorat	A-eksperimental	B-eksperimental
Ichki motivatsiya (1-5)	2.9	3.7	4.1
O‘z-o‘ziga ishonch	3.1	3.9	4.2
Darsga qatnashish (%)	78%	89%	93%
Mustaqil loyiha yaratish	12%	38%	47%
Kasbiy rivojlanishga qiziqish	54%	79%	84%

B-eksperimental guruhda talabalarning 47% i semestr davomida o‘quv dasturidan tashqari mustaqil loyiha yaratdi — bu OOP ni chuqur tushunganliklarining bevosita ko‘rsatkichidir.

3.6. Korrelyatsiya tahlili: Pearson korrelyatsiya tahlili OOP bilim testi natijalari bilan quyidagi o‘zgaruvchilar o‘rtasidagi bog'liqlikni aniqladi:

- Amaliy mashg'ulotlardagi faollik: $r = 0.71$ ($p < 0.001$) — kuchli musbat bog'liqlik
- Real loyihalar soni: $r = 0.68$ ($p < 0.001$) — kuchli musbat bog'liqlik
- O‘quv motivatsiyasi: $r = 0.63$ ($p < 0.001$) — o‘rtacha-kuchli musbat bog'liqlik
- Leksiyaga qatnashish: $r = 0.34$ ($p < 0.01$) — zaif musbat bog'liqlik

Bu natija ayniqsa muhim: leksiyaga qatnashish OOP bilimini faqat zaif

Latin American Journal of Education
www.lajoe.org

darajada bashorat qiladi, amaliy faollik esa kuchli omil hisoblanadi.

MUHOKAMA (Discussion)

4.1. Asosiy topilmalar tafsiri: Tadqiqotning eng muhim topilmasi — "Objects First" va gibrid yondashuvlar an'anaviy protsedurali-birinchi (procedural-first) yondashuvga nisbatan sezilarli darajada samaraliroq ekanligi. Bu natija Kölling va Rosenbergning (2001) BlueJ asosidagi tadqiqotlari va Barnes & Kölling (2009) tomonidan ommalashtirgan "Objects First with Java" kontseptsiyasini tasdiqlaydi.

Gibrid yondashuvning (B-guruh) barcha koʻrsatkichlarda biroz ustunligi bir muhim narsani koʻrsatadi: dasturlash tilining sintaksik soddaliği OOP tushunchalarini oʻzlashtirish uchun "kognitiv joy" ochib beradi. Python ning sodda sintaksisi talabaning barcha e'tiborini OOP mantiqiga qaratishga yordam beradi.

4.2. Kognitiv yuklanma nazariyasi nuqtai nazaridan tahlil: Sweller (1988) CLT nazariyasi nuqtai nazaridan qarasak, an'anaviy yondashuv talabaga bir vaqtning oʻzida juda koʻp narsa oʻrgatadi: avval protsedurali dasturlash (uning oʻzi katta kognitiv yuklanma), soʻngra butunlay boshqacha paradigma — OOP. Natijada talaba OOP ga "charchagan holda" keladi.

"Objects First" va gibrid yondashuvlar esa kognitiv yuklanmani boshqacha taqsimlaydi. B-guruhda Python ning sodda sintaksisi orqali avval dasturlash mantig'i oʻzlashtiriladi, OOP esa asta-sekin, real kontekstda kiritiladi. Bu "scaffolding" (qoʻllab-quvvatlash) tamoyiliga toʻliq mos keladi.

4.3. OOP oʻqitishning optimal ketma-ketligi: Tadqiqot natijalari va adabiyotlar tahlili asosida OOP oʻqitishning quyidagi ketma-ketligi tavsiya etiladi:

1-bosqich (2-3 hafta): Ob'ekt nima? — Mavhum tushunchadan emas, real dunyo ob'ektlaridan boshlash. Mashina, kitob, talaba — bularning xususiyatlari (attributes) va xatti-harakatlari (methods) nima? Bu bosqichda kod yozilmaydi, faqat tushuncha shakllantiriladi.

2-bosqich (3-4 hafta): Oddiy sinflar — Bitta sinf, bir necha atribut va metod. Kapsulalash, getter/setter. Real misollar: BankAccount, Student, Book sinflari.

3-bosqich (2-3 hafta): Ob'ektlar oʻrtasidagi munosabat — Kompozitsiya (has-a munosabati). Library va Book, Car va Engine. Merosdan oldin kompozitsiyani oʻrgatish muhim, chunki amaliyotda kompozitsiya koʻproq ishlatiladi.

4-bosqich (3-4 hafta): Meros va polimorfizm — Faqat meros mantiqiy boʻlgan hollarda: Shape → Circle/Rectangle, Animal → Dog/Cat. Polimorfizmni loyiha orqali oʻrgatish.

5-bosqich (2-3 hafta): Abstraktsiya va interfeyslar — Abstract class va interface farqi. Real loyihada qoʻllash.

6-bosqich (3-4 hafta): Dizayn naqshlari asoslari — Singleton, Factory, Observer. Bu bosqichda talabalar OOP ning "nima uchun" savoliga toʻliq javob topa boshlaydi.

Latin American Journal of Education
www.lajoe.org

4.4. O‘qituvchi roli va metodologik yondashuv: Tadqiqot davomida kuzatuv shuni ko‘rsatdiki, eng samarali o‘qituvchilar quyidagi xususiyatlarga ega edi: ular "kod nima qiladi" emas, "kod nima muammoni hal qiladi" degan savoldan boshladilar; xatolarni jamoat oldida emas, shaxsiy ko‘rsatuvchi tarzda tuzatdilar; har bir yangi tushunchani oldingi bilan aniq bog'ladilar; real sanoat misollari keltirdilar.

Bundan tashqari, "rubber duck debugging" — kodni xayoliy o‘rdakka tushuntirish — metodidan foydalanish talabalarning o‘z xatolarini topishiga yordam berdi va mustaqil fikrlash ko‘nikmalarini rivojlantirdi.

4.5. Raqamli vositalar va OOP o‘qitish: Tadqiqot davomida qo‘llanilgan raqamli vositalarning samaradorligi alohida baholandi:

BlueJ — OOP tushunchalarini vizual tarzda ko‘rsatish uchun juda samarali. Ob'ektlar qutichalar sifatida ko‘rinadi, ularning munosabatlari strelkalar bilan ifodalanadi. Yangi boshlovchilar uchun ideal.

IntelliJ IDEA — professional muhit, avtoto‘ldirish va refactoring vositalari OOP tuzilmasini tushunishga yordam beradi. 3-4-bosqichdan foydalanish maqsadga muvofiq.

UML diagrammalari (draw.io, PlantUML) — sinf diagrammalarini chizish OOP arxitekturasini vizualizatsiya qilishda samarali. Loyiha boshlashdan oldin UML chizish odatini shakllantirish muhim.

Codewars, LeetCode — OOP masalalarini yechish uchun foydali, lekin ular ko‘proq algoritmga yo‘naltirilgan. OOP uchun maxsus platforma — Exercism.io — ancha mos keladi.

4.6. Tadqiqot cheklovlari: Ushbu tadqiqotning bir qator cheklovlari mavjud. Birinchidan, tadqiqot faqat ikkita universitetda o‘tkazildi va natijalar boshqa kontekstlarga to‘liq ko‘chirilmasligi mumkin. Ikkinchidan, o‘qituvchi effekti (teacher effect) to‘liq nazorat qilinmagan — eksperimental guruhlarga yanada innovatsion o‘qituvchilar tayinlangan bo‘lishi mumkin. Uchinchidan, uzoq muddatli ta'sir (longitudinal effect) o‘rganilmagan: talabalar ish faoliyatida OOP ni qanday qo‘llashlari kuzatilmadi.

Kelajakdagi tadqiqotlar uchun bitiruvchilarni 2-3 yil kuzatib borish, ko‘proq universitetlarni jalb etish va sun'iy intellekt vositalarining OOP o‘qitishga ta'sirini o‘rganish tavsiya etiladi.

XULOSA VA TAVSIYALAR

5.1. Asosiy xulosalar: Ushbu tadqiqot quyidagi muhim xulosalarga olib keldi:

Birinchi xulosa: OOP ni dasturlash o‘qitishning dastlabki bosqichidanoq kiritish (Objects First yondashuvi) talabalarning OOP tushunchalarini o‘zlashtirish darajasini an'anaviy yondashuvga nisbatan o‘rtacha 13-17 foizga oshiradi.

Ikkinchi xulosa: Sintaksisi oddiy til (Python) orqali OOP o‘rgatish kognitiv yuklanmani kamaytiradi va ayniqsa

Latin American Journal of Education
www.lajoe.org

dasturlash bilan birinchi marta tanishayotgan talabalar uchun samaraliroq.

Uchinchi xulosa: OOP o‘qitishda amaliy faollik (r = 0.71) leksiyaga qatnashishdan (r = 0.34) ancha kuchli omil hisoblanadi. Bu dasturlash fanlarida auditoriya vaqtini qanday taqsimlash kerakligini qayta ko‘rib chiqish zarurligini ko‘rsatadi.

To‘rtinchi xulosa: OOP tushunchalarini o‘rgatishning optimal ketma-ketligi — ob'ekt → kapsulalash → kompozitsiya → meros → polimorfizm → abstraktsiya → dizayn naqshlari — an'anaviy ketma-ketlikdan samaraliroq.

5.2. Amaliy tavsiyalar:

O‘qituvchilarga:

- Har bir OOP darsining kamida 60% ini amaliy kodlashga ajrating
- Yangi tushunchani o‘rgatishda avval real dunyo analogiyasini keltiring, so‘ngra koddagi ifodasini ko‘rsating
-
- Talabalarni juft dasturlash (pair programming) orqali ishlashga muntazam jalb eting
- Xatolardan qo‘rqmaslikka o‘rgating — live coding da o‘qituvchining o‘zi ham xato qilib tuzatishi talabaga katta psixologik qulaylik beradi
- UML diagrammasini har bir loyiha oldidan majburiy qildirib chizing

O‘quv muassasalariga:

- Dasturlash fanlarini o‘qitishda "Objects First" yoki gibrid yondashuvga o‘tishni ko‘rib chiqing
- O‘qituvchilarni zamonaviy OOP o‘qitish metodologiyalari bo‘yicha muntazam malaka oshirishga yuboring
- BlueJ va boshqa vizual muhitlarni dasturlash kurslarining dastlabki bosqichlarida joriy eting
- Sanoat mutaxassislarini darsga taklif qilish va real loyihalar bilan ishlash mexanizmlarini yarating

FOYDALANILGAN ADABIYOTLAR:

1. Barnes, D. J., & Kölling, M. (2009). *Objects First with Java: A Practical Introduction Using BlueJ* (4th ed.). Prentice Hall.
2. Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming. *ACM SIGCSE Bulletin*, 39(2), 32–36.
3. Freeman, S., et al. (2014). Active learning increases student performance in science, engineering, and mathematics. *PNAS*, 111(23), 8410–8415.
4. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
5. Kölling, M. (2008). The Greenfoot programming environment. *ACM Transactions on Computing Education*, 10(4), 1–21.

Latin American Journal of Education
www.lajoe.org

6. Kölling, M., & Rosenberg, J. (2001). Guidelines for teaching object orientation with Java. *ACM SIGCSE Bulletin*, 33(3), 33–36.

7. Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137–172.

8. Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2), 257–285.

9. TIOBE Index (2024). TIOBE Programming Community Index. Manba: tiobe.com/tiobe-index

10. Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.

11. Xolmatov, S., & Yusupov, B. (2023). Oliy ta'limda OOP o‘qitishning zamonaviy muammolari. *TATU Ilmiy axborotnomasi*, 2(1), 44–53.

12. Yo‘ldoshev, A. (2022). Dasturlash fanlarida faol o‘qitish metodlarining samaradorligi. *Pedagogik mahorat*, 4, 98–107.

13. Karimov, N., & Toshmatov, F. (2023). Python va Java orqali OOP o‘qitish: qiyosiy tahlil. *O‘zbekiston ta'lim jurnali*, 5(3), 61–72.

14. Stack Overflow (2023). *Developer Survey 2023*. stackoverflow.com/survey/2023

15. Exercism.io (2024). Object-Oriented Programming Track Documentation. exercism.org