MASHINA KODLARI BILAN ISHLASH

Tojimamatov Israiljon Nurmamatovich
Farg‘ona davlat universiteti katta o‘qituvchisi
israiltojimamatov@gmail.com
Meliqo‘ziyev Sahobiddin Umidjon o‘g‘li
Farg‘ona davlat universiteti 2-bosqich talabasi
sahobiddinmeliqoziyev1@gmail.com

Anotatsiya: *Ushbu maqolada kod tushunchasining nazariy va amaliy jihatlari, uning tizimli dasturlashdagi roli, mashina kodlari va assembler tilidagi ifodalanishi yoritiladi. Shuningdek, kod tuzish jarayoni, optimallashtirish, va kompyuter arxitekturasi bilan bog‘liqligi tahlil qilinadi. Tizimli dasturlash jarayonida kodni samarali va barqaror yozish usullari, kompilyatsiya va interpretatsiya bosqichlari, hamda kod xavfsizligi masalalari haqida ma’lumotlar keltirilgan. Maqola talaba va dasturchilar uchun amaliy foydalanish imkoniyatlarini kengaytirishga xizmat qiladi.*

Kalit so'zlar: *Kod, tizimli dasturlash, assembler, mashina kodi, kompilyator, interpretator, dasturiy ta'minot, operatsion tizim, registrlar, bayt kodi, instruktsiyalar, kompyuter arxitekturasi, dastur optimallashtirish, past darajadagi dasturlash, kod xavfsizligi*

Abstract: *This article explores the theoretical and practical aspects of the concept of code and its role in system programming. It examines machine code representation, the use of assembly language, code structure and optimization processes, and its relationship with computer architecture. The study also discusses effective and stable coding practices in system-level programming, compilation and interpretation stages, as well as code security considerations. The*

article aims to enhance the practical understanding of students and developers in low-level programming.

Keywords: *Code, system programming, assembler, machine code, compiler, interpreter, software, operating system, registers, bytecode, instructions, computer architecture, code optimization, low-level programming, code security*

Аннотация: *В данной статье рассматриваются теоретические и практические аспекты понятия кода и его роль в системном программировании. Описываются представление машинного кода, использование языка ассемблера, структура и оптимизация кода, а также его связь с архитектурой компьютера. Также анализируются методы эффективного и стабильного программирования на системном уровне, этапы компиляции и интерпретации, и вопросы безопасности кода. Статья предназначена для расширения практических знаний студентов и разработчиков в области низкоуровневого программирования.*

Ключевые слова: *код, системное программирование, ассемблер, машинный код, компилятор, интерпретатор, программное обеспечение, операционная система, регистры, байткод, инструкции, архитектура компьютера, оптимизация кода, низкоуровневое программирование, безопасность кода*

Zamonaviy texnologiyalar taraqqiyoti bilan dasturlash sohasining ahamiyati keskin ortib bormoqda. Har qanday dastur yoki raqamli tizimning asosi bo‘lgan kod — bu kompyuter bilan inson o‘rtasidagi muloqot vositasi hisoblanadi. Oddiy qilib aytganda, kod — bu buyruqlar to‘plami bo‘lib, u orqali inson kompyuterga nima qilish kerakligini ko‘rsatadi. Kod yozish orqali turli dasturlar, operatsion tizimlar, mobil ilovalar va hatto sun’iy intellekt tizimlari yaratiladi. Dasturlashdagi eng asosiy jarayon aynan kod yozishdan iborat bo‘lib, bu soha o‘zida matematik mantiq, algoritmik fikrlash, va texnik tafakkurni birlashtiradi.

Kod tushunchasi dastlab faqat mashina darajasidagi buyruqlar shaklida mavjud bo‘lgan. Ya’ni, dasturchilar kompyuterning to‘g‘ridan-to‘g‘ri tushunadigan 0 va 1 lar ketma-ketligini — binar kodni yozish orqali dastur yaratishgan. Keyinchalik bu jarayon soddalashtirilib, assembler kabi past darajadagi dasturlash tillari paydo bo‘ldi. Bu tillar orqali kod yozish nisbatan qulaylashgan bo‘lsa-da, kompyuter texnikasi va arxitekturasi haqida chuqur bilim talab qilardi. Hozirgi kunda esa yuqori darajadagi dasturlash tillari orqali inson tushunadigan sintaksisda kod yozish mumkin bo‘lib, bu dasturlashni keng ommaga ochib berdi.

Kod yozish nafaqat texnik ko‘nikma, balki ijodiy jarayon hamdir. Dasturchi har bir satrni puxta o‘ylab, tizimga qanday ta’sir ko‘rsatishini hisobga olishi zarur. Shuning uchun kod yozish madaniyati, ya’ni toza, tushunarli va strukturaviy jihatdan to‘g‘ri kod yozish, dasturchining asosiy vazifalaridan biri hisoblanadi. Bugungi kunda kodni samarali yozish, uni testlash va optimallashtirish dasturlashning ajralmas qismiga aylangan. Umuman olganda, kod tushunchasi dasturlashning yuragi bo‘lib, har bir dasturiy mahsulotning negizini tashkil etadi. Kodni chuqur o‘rganish orqali nafaqat yangi texnologiyalarni yaratish, balki mavjud tizimlarni ham samarali boshqarish mumkin. Shu bois kod yozishni o‘rganish — bu zamonaviy texnologiyalar olamiga kirish eshigi hisoblanadi.

Kod dasturlash tilining asosiy ifoda shakli bo‘lib, u turli darajalarda kompyuter bilan muloqot qilish imkonini beradi. Umuman olganda, kodlar ikki asosiy turga — **yuqori darajadagi** va **past darajadagi** kodlarga bo‘linadi. Har bir tur o‘zining abstraktsiya darajasi, tushunarlilik darajasi hamda kompyuter texnikasiga yaqinligi bilan farqlanadi.

Yuqori darajadagi kod (high-level code) inson tili va mantiqiga yaqin bo‘lgan dasturlash tillarida yoziladi. Bu turdagi kodlar C#, Java, Python, va boshqa shu kabi tillarda yozilgan bo‘lib, ular kompilyator yoki interpretator yordamida past darajadagi kodga aylantiriladi. Yuqori darajadagi kodlarning asosiy afzalligi

— ular o‘qishga qulay, strukturali va ko‘proq algoritmik ifodani o‘zida mujassam etadi. Dasturchilar bu turdagi tillardan foydalanib, murakkab dasturlarni tez va samarali yaratish imkoniyatiga ega bo‘ladilar. Biroq bu kodlar bevosita kompyuter protsessori tomonidan tushunilmaydi, ularni avval kompilyatsiya yoki interpretatsiya qilish zarur.

Past darajadagi kod (low-level code) esa kompyuter arxitekturasiga yaqin bo‘lgan ko‘rsatmalar to‘plamidan iborat. Bu yerda assembler tili alohida o‘rin tutadi. Assemblerda yozilgan kodlar protsessorning registrlariga, xotira adreslariga va boshqa texnik detallariga bevosita murojaat qiladi. Shu sababli, past darajadagi kodlar yuqori darajadagi kodlarga nisbatan murakkab, lekin aniqroq va samaraliroq ishlaydi. Masalan, real vaqtda ishlovchi tizimlar, drayverlar yoki operatsion tizim yadrosi kabi dasturlar aynan past darajadagi kod asosida yaratiladi. Dasturchi bu yerda kompyuterning "ichki dunyosi" bilan bevosita ishlaydi.

Shuningdek, kodlarning eng quyi shakli bu **mashina kodi** hisoblanadi. Bu kod faqat ikkilik (binary) sonlardan — 0 va 1 lar ketma-ketligidan tashkil topgan bo‘lib, bevosita kompyuter protsessori tomonidan bajariladi. Mashina kodi inson uchun tushunarsiz bo‘lsa-da, u barcha dasturlar uchun yakuniy bajariladigan shakldir. Masalan, assembler tilida yozilgan kod kompilyatsiyadan so‘ng aynan mashina kodiga aylantiriladi. Zamonaviy dasturlashda shuningdek, **bayt kodi** (bytecode) kabi oraliq shakllar ham qo‘llaniladi. Masalan, Java dasturlash tilida yozilgan dastur avval bayt kodga aylantiriladi va so‘ngra Java Virtual Machine (JVM) tomonidan bajariladi. Bu yondashuv platformadan mustaqil ishlash imkoniyatini beradi.Kod turlari kompyuter bilan ishlash darajasiga qarab farqlanadi. Yuqori darajadagi kod inson uchun qulay va samarali bo‘lsa, past darajadagi kod tizimni chuqur nazorat qilish imkonini beradi. Har bir turdagi kodning o‘ziga xos afzallik va kamchiliklari mavjud bo‘lib, ular dasturiy mahsulotning turiga, talablariga va ishlash muhitiga qarab tanlanadi.

Kod yozish bu faqatgina buyruqlar ketma-ketligini kompyuterga kiritish emas, balki bu murakkab fikrlash, rejalashtirish va strukturaviy yondashuvni talab qiladigan jarayondir. Har bir dasturchi kod yozishda ma’lum tamoyillar va qoidalarga amal qilishi zarur. Bu tamoyillar kodni nafaqat to‘g‘ri ishlaydigan, balki tushunarli, kengaytiriladigan va xizmat ko‘rsatish oson bo‘lgan holatda yaratishga yordam beradi. Eng avvalo, kod yozishda **sintaksis** va **semantika** tushunchalariga e’tibor qaratish lozim. Sintaksis — bu dasturlash tilining grammatik qoidalari bo‘lib, ularni buzgan holda yozilgan kodlar kompilyator yoki interpretator tomonidan tan olinmaydi. Semantika esa yozilgan kodning ma’nosini anglatadi, ya’ni kod nimani bajaradi va qanday natija beradi degan savolga javob beradi. Kod yozishda bu ikki omil birgalikda muhim ahamiyatga ega bo‘lib, dastur muvaffaqiyatli ishlashi uchun ularning to‘g‘ri qo‘llanilishi talab etiladi.

Kod yozishda **strukturaviylik** tamoyili ham alohida o‘rin tutadi. Strukturaviy kod — bu aniq bloklarga bo‘lingan, funksiyalar yoki modullar orqali tuzilgan va o‘qishga qulay bo‘lgan koddir. Bu yondashuv orqali dastur ichidagi har bir qismning vazifasi aniq ajratiladi, bu esa kodni sinovdan o‘tkazish, tuzatish yoki keyinchalik kengaytirish ishlarini ancha yengillashtiradi. Aynan shu sababli funksiyalar, sinflar, interfeyslar va modullar yordamida kodni organizatsiya qilish amaliy dasturlashda keng qo‘llaniladi. Bundan tashqari, kod yozishda **tozalik** (clean code) tamoyili muhim hisoblanadi. Bu tamoyilga ko‘ra, kod tushunarli nomlar, izohlar (kommentariyalar), ortiqcha takrorlanmagan va mantiqiy ketma-ketlikda yozilishi kerak. Dasturchi kod yozayotganda uni boshqa dasturchi ham o‘qib, tushunishi mumkin bo‘lishi kerak. Kodga izohlar qo‘shish, har bir blok nima uchun yozilganini tushuntirish bu jihatdan foydali bo‘ladi. Lekin haddan ortiq izohlar emas, balki zarur joylarda, maqsadga muvofiq tarzda izoh berish tavsiya etiladi.

Kod yozish jarayonida **xatoliklarni aniqlash va ularni bartaraf etish** ham muhim bosqich hisoblanadi. Har qanday dastur birinchi yozilishidayoq ideal

ishlamaydi. Unda sintaktik, mantiqiy yoki ishlashga oid (runtime) xatoliklar bo‘lishi mumkin. Shuning uchun dasturchi kodni sinovdan o‘tkazish, ya’ni testlash orqali ushbu muammolarni aniqlashi va to‘g‘rilashi kerak. Avtomatlashtirilgan testlar, log yozuvlari va xatoliklarni qayd etuvchi vositalar kod sifatini oshirishda muhim rol o‘ynaydi. Yana bir muhim tamoyil — bu **DRY (Don't Repeat Yourself)** qoidasi bo‘lib, u kodda ortiqcha takroriylikka yo‘l qo‘ymaslikni tavsiya qiladi. Takroriy kodni alohida funksiyaga yoki modulga chiqarish orqali kodni ixcham, boshqarilishi oson va samarali qilish mumkin. Bu nafaqat kodni o‘qish va tushunishni osonlashtiradi, balki uni kelajakda o‘zgartirishda ham qulaylik yaratadi.

Kod va kompyuter tizimlari bir-biriga uzviy bog‘liq bo‘lgan tushunchalardir. Dasturlashda yozilgan har bir kod satri bevosita kompyuter tizimining qanday ishlashiga ta’sir ko‘rsatadi. Kod bu yerda buyruqlar majmuasi sifatida harakat qilib, kompyuter resurslarini boshqaradi: xotira, protsessor, kiritish-chiqarish qurilmalari va tarmoqlar orqali amalga oshiriladigan barcha jarayonlar aynan kod orqali nazorat qilinadi. Boshqacha aytganda, kod — bu kompyuterning barcha tizimli qismlariga murojaat qiluvchi til bo‘lib, u orqali inson texnik qurilmalarga aniq vazifalarni topshiradi.

Kod yozilgan dasturlar to‘g‘ridan-to‘g‘ri kompyuter operatsion tizimi bilan aloqada bo‘ladi. Masalan, Windows, Linux yoki macOS kabi operatsion tizimlar dasturning bajarilishi, xotira taqsimoti, fayl tizimlariga kirish va boshqa tizim resurslarini boshqarish bo‘yicha o‘ziga xos API (Application Programming Interface) larni taqdim etadi. Dasturchi bu interfeyslar orqali o‘zining yozgan kodi orqali kompyuter tizimining ichki mexanizmlariga kirish imkoniyatiga ega bo‘ladi. Bu esa, kodning faqatgina hisob-kitob emas, balki tizim darajasidagi boshqaruv uchun ham javobgar ekanligini anglatadi. Kodning kompyuterda bajarilishi ikki asosiy bosqich orqali amalga oshadi: **kompilyatsiya** va **interpretatsiya**. Yuqori darajadagi dasturlash tillarida yozilgan kodlar dastlab kompilyator yoki

interpretator yordamida mashina kodiga aylantiriladi. Kompilyatsiya — bu dastur matnini birdaniga to‘liq bajariladigan mashina kodiga aylantiruvchi jarayon bo‘lsa, interpretatsiya esa kodni qadam-baqadam bajarib boruvchi yondashuvdir. Har ikkala holatda ham, dastur oxir-oqibatda kompyuter protsessori tushunadigan ko‘rinishga keltiriladi. Bu jarayonlarda operatsion tizim vositachi rolini bajaradi va dasturga kerakli tizim resurslarini taqdim etadi. Kod kompyuter tizimlari bilan **xavfsizlik** jihatidan ham bog‘liq. Noto‘g‘ri yozilgan yoki zararli kod tizimdagi ma’lumotlarni o‘g‘irlashi, zarar yetkazishi yoki tizimni izdan chiqarishi mumkin. Shu sababli, tizimli dasturlashda kodni xavfsiz yozish, resurslardan to‘g‘ri foydalanish va foydalanuvchi huquqlariga hurmat bilan qarash juda muhim hisoblanadi. Operatsion tizimlar bu borada kod bajarilishini nazorat qiluvchi turli mexanizmlarga ega: fayl tizimi ruxsatnomalari, sand-boxlash texnologiyalari, antivirus tizimlari va boshqa himoya qatlamlari orqali noto‘g‘ri yoki zararli kodlarning tizimga zarar yetkazishining oldi olinadi.

Kod va kompyuter tizimlari o‘rtasidagi yana bir muhim bog‘liqlik bu — **resurslarni optimallashtirish** masalasidir. Har bir dastur, albatta, tizim resurslaridan foydalanadi: operativ xotira, protsessor va doimiy xotira (disk) kabi. Yaxshi yozilgan kod ushbu resurslardan samarali foydalanadi, ortiqcha yuklamaga olib kelmaydi va tizimni sekinlashtirmaydi. Aksincha, noto‘g‘ri yozilgan kodlar — xotira sizib chiqishi (memory leak), cheksiz sikllar, yoki optimal bo‘lmagan algoritmlar orqali tizim samaradorligini keskin pasaytirishi mumkin. Shu sababli, kod yozishda tizim arxitekturasi, apparat imkoniyatlari va ishlash muhiti hisobga olinishi shart.

Xulosa qilib aytganda tizimli dasturlash bugungi kunda dasturiy ta'minotni ishlab chiqish, testlash va qo‘llash jarayonlarining asosi hisoblanadi. Kod — bu kompyuter tizimlari bilan muloqot qilishning yagona vositasi bo‘lib, uning har bir satri muayyan vazifani bajaradi va tizimda aniq o‘zgarishlarni yuzaga keltiradi. Ushbu maqolada kodning mohiyati, turlari, uni yozish tamoyillari hamda

kompyuter tizimlari bilan qanday bog‘liq ekani batafsil yoritildi. Ko‘rib chiqilgan barcha jihatlar shuni ko‘rsatadiki, kod — bu nafaqat matematik yoki texnik vosita, balki murakkab tizimlar bilan ishlashni boshqaruvchi asosiy mexanizmdir.

Kod turlari — yuqori darajadagi, past darajadagi va mashina kodi — o‘zaro bog‘langan bo‘lib, har biri muayyan vazifalarni bajarishda qo‘llaniladi. Dastur samaradorligi, tizimga yuk tushishi, xavfsizlik darajasi va foydalanuvchi qulayligi aynan qanday kod yozilganiga bog‘liq. Kod yozish tamoyillariga amal qilish, ya’ni strukturaviylik, tozalik, DRY va testlash kabi yondashuvlar dastur sifatini oshiradi. Kod va kompyuter tizimlari o‘rtasidagi bog‘liqlik esa bizga kod orqali texnik resurslarni boshqarish, samaradorlikni ta’minlash va xavfsizlikni nazorat qilish imkonini beradi.

Shu bilan birga, tizimli dasturlash sohasi nafaqat texnik bilimlarni, balki mantiqiy tafakkurni, tizimlarni chuqur tushunishni va mas'uliyatli yondashuvni talab qiladi. Har bir yozilgan kod satri ortida tizimning mukammal ishlashi, foydalanuvchi ehtiyojlarini qondirish va texnologik taraqqiyot yotadi. Shu bois, kod yozish jarayonini chuqur o‘rganish va tizimli yondashuvni o‘zlashtirish har bir zamonaviy dasturchining ustuvor vazifalaridan biridir.

Foydalanilgan adabiyotlar ro’yhati:

1. Tanenbaum A. S. — "Structured Computer Organization", Pearson Education.
2. Stallings W. — "Computer Organization and Architecture", Pearson.
3. Patterson D. A., Hennessy J. L. — "Computer Organization and Design", Morgan Kaufmann.
4. Kip R. Irvine — "Assembly Language for x86 Processors", Pearson.
5. Andrew S. Tanenbaum — "Modern Operating Systems", Prentice Hall.
6. Charles Petzold — "Code: The Hidden Language of Computer Hardware and Software".
7. Brian W. Kernighan, Dennis M. Ritchie — "The C Programming Language".

8. Jeff Duntemann — "Assembly Language Step-by-Step: Programming with Linux".

9. Randall Hyde — "The Art of Assembly Language", No Starch Press.

10. Silberschatz A., Galvin P. B., Gagne G. — "Operating System Concepts".

11. Nisan N., Schocken S. — "The Elements of Computing Systems: Building a Modern Computer from First Principles".

12. Steve McConnell — "Code Complete", Microsoft Press.

13. Michael L. Scott — "Programming Language Pragmatics".

14. William Shotts — "The Linux Command Line".

15. Maurice J. Bach — "The Design of the UNIX Operating System".

16. Tojmamatov, I., & Usmonjon o'g'li, S. I. (2025). GRAFIK AXBOROTLARNI KODLASH, ULARNI QAYTA ISHLASH. *JOURNAL OF INTERNATIONAL SCIENTIFIC RESEARCH*, *2*(2), 203-207.

17. Nurmamatovich, I. T., & Usmonjon o'g'li, S. I. (2025). MASHINA AMALLARI; ARIFMETIK-MANTIQIY, JONALISH, OTISH, KIRITISH-CHIQARISH VA SISTEMA AMALLARI. *JOURNAL OF INTERNATIONAL SCIENTIFIC RESEARCH*, *2*(2), 155-159.

18. Nurmamatovich, T. I. (2025). KODLASHNING NAZARIY ASOSLARI. *BILGI ÇEŞMESI*, *1*(3), 63-72.

19.Nurmamatovich, T. I. (2025). ASOSIY ALGORITMIK TUZULMALAR. *СОВРЕМЕННОЕ ОБРАЗОВАНИЕ И ИССЛЕДОВАНИЯ*, *1*(3), 68-70.

20. Tojmamatov, I., & Usmonjon o'g'li, S. I. (2025). GRAFIK AXBOROTLARNI KODLASH, ULARNI QAYTA ISHLASH. *JOURNAL OF INTERNATIONAL SCIENTIFIC RESEARCH*, *2*(2), 203-207.

21. Nurmamatovich, T. I. (2025). MATNLI AXBOROTLARNI TURLI USULLARDA KODLASH. *JOURNAL OF INTERNATIONAL SCIENTIFIC RESEARCH*, *2*(2), 191-202.

22. Nurmamatovich, T. I., & Usmonjon o'g'li, S. I. (2025). MA’LUMOTLAR QIDIRISHDA KETMA-KET QIDIRUV. IKKILIK QIDIRUVI. *YANGI O ‘ZBEKISTON, YANGI TADQIQOTLAR JURNALI*, *2*(2), 36-42.

23. Nurmamatovich, T. I., & Usmonjon o'g'li, S. I. (2025). MA’LUMOTLAR QIDIRISHDA KETMA-KET QIDIRUV. IKKILIK QIDIRUVI. *YANGI O ‘ZBEKISTON, YANGI TADQIQOTLAR JURNALI*, *2*(2), 36-42.

a