ISSN 2181-3187

DASTURLARNI BAJARILISHI

Tojimamatov Israiljon Nurmamatovich
Farg‘ona davlat universiteti katta o‘qituvchisi
israiltojimamatov@gmail.com
Ismoilov Javohir Ulug‘bek o‘g‘li
Farg‘ona davlat universiteti 2-bosqich talabasi
javohir20060612@gmail.com

Anotatsiya: Ushbu maqolada dasturlarni bajarilishi jarayoni, ya’ni dasturiy kod yozilgandan so‘ng u qanday qilib bajarilishini tushunishning nazariy va amaliy asoslari yoritilgan. Dasturlarni bajarish bosqichlari, kompilatsiya va interpretatsiya usullari, virtual mashinalarning roli, operatsion tizim bilan o‘zaro aloqasi hamda zamonaviy texnologiyalar (masalan, JIT — Just-In-Time compilation) asosida dastur bajarilishining samaradorligi tahlil qilingan. Maqolada turli dasturlash tillarida bajarilish mexanizmlarining farqlari, samaradorlik darajalari va xavfsizlik jihatlari ham ko‘rib chiqiladi.

Kalit so'zlar: Dastur bajarilishi, kompilatsiya, interpretatsiya, JIT, virtual mashina, bajarilish muhiti, kod optimallashtirish, operatsion tizim, dasturlash tillari, bytecode, runtime, debugger, assembler, kompilyator, interpreter

Annotation: This article explores the process of program execution, focusing on both theoretical and practical aspects of how code is executed after being written. It discusses the stages of execution, compilation and interpretation methods, the role of virtual machines, interaction with the operating system, and the effectiveness of modern technologies such as Just-In-Time (JIT) compilation. Differences in execution mechanisms among various programming languages, performance considerations, and security aspects are also analyzed.

Keywords: Program execution, compilation, interpretation, JIT, virtual machine, execution environment, code optimization, operating system, programming languages, bytecode, runtime, debugger, assembler, compiler, interpreter

ISSN 2181-3187

Аннотация: В данной статье рассматривается процесс выполнения программ, включая теоретические и практические аспекты выполнения кода после его написания. Освещаются этапы выполнения, методы компиляции и интерпретации, роль виртуальных машин, взаимодействие с операционной системой, а также эффективность современных технологий, таких как компиляция Just-In-Time (JIT). Также анализируются различия в механизмах выполнения среди различных языков программирования, вопросы производительности и безопасности.

Ключевые слова: Выполнение программ, компиляция, интерпретация, JIT, виртуальная машина, среда выполнения, оптимизация кода, операционная система, языки программирования, байт-код, выполнение, отладчик, ассемблер, компилятор, интерпретатор

Zamonaviy dunyoda dasturlash texnologiyalari kundalik hayotimizning ajralmas qismiga aylanib bormoqda. Mobil ilovalar, veb-saytlar, avtomatlashtirilgan tizimlar va hatto sun’iy intellekt asosida ishlaydigan qurilmalar – bularning barchasi dasturiy ta’minotlar orqali boshqariladi. Har qanday dastur ishlashi uchun u yozilgan kod asosida bajarilishi kerak. Shuning uchun ham “dasturlarni bajarilishi” mavzusi informatika va dasturlash sohasida asosiy tushunchalardan biri hisoblanadi. Bu jarayon dasturchi tomonidan yozilgan kodning kompyuter tomonidan qanday tushunilib, qanday tartibda bajarilishi haqida tasavvur hosil qilishga yordam beradi.

Dasturlarni bajarilishi bu faqatgina oddiy kodni ishga tushurish emas, balki u orqasida yotgan murakkab texnologik jarayonlarni anglatadi. Har bir dastur yozilgan dasturlash tiliga qarab turlicha yo‘llar bilan bajarilishi mumkin. Masalan, ba’zi dasturlar avval kompilyatsiya qilinadi, ya’ni yuqori darajadagi dasturlash tili mashina tiliga tarjima qilinadi, so‘ngra bajariladi. Boshqa dasturlar esa to‘g‘ridan-to‘g‘ri interpretatsiya qilinadi, ya’ni har bir buyruq ketma-ket tarzda o‘qilib, bajariladi. Shu bilan birga, zamonaviy texnologiyalar, masalan, Just-In-Time (JIT) kompilyatsiya, dastur bajarilishini yanada samarali va tez amalga oshirish imkonini beradi.

ISSN 2181-3187

Mazkur maqolada dasturlarni bajarilishi bilan bog‘liq asosiy tushunchalar, bajarilish jarayonining bosqichlari, kompilatsiya va interpretatsiya texnologiyalari, virtual mashinalarning roli, hamda amaliy dasturlashda bu bilimlarning qanday ahamiyatga ega ekani atroflicha yoritiladi. Mavzuni chuqur tushunish dasturchilarga nafaqat samarali dastur yozish, balki ularning ish faoliyatini optimallashtirish, resurslardan oqilona foydalanish va tizim xavfsizligini ta'minlashda ham muhim ahamiyat kasb etadi.

Dastur bajarilishi – bu foydalanuvchi yoki tizim tomonidan ishga tushirilgan dastur kodining kompyuter tomonidan izchil tarzda bajarilishi jarayonidir. Bu jarayon bir nechta asosiy bosqichlardan iborat bo‘lib, har bir bosqich dastur ishlashining muhim qismidir. Avvalo, dasturchi dastur kodini yuqori darajadagi dasturlash tilida yozadi. Ushbu kod inson o‘qiy oladigan shaklda bo‘ladi, lekin kompyuter uni to‘g‘ridan-to‘g‘ri tushunmaydi. Shu sababli dastlab kodni mashina tushunadigan formatga aylantirish zarur bo‘ladi. Bu bosqichda kompilatsiya yoki interpretatsiya jarayoni ishga tushadi.

Kompilyator yordamida bajariladigan tillarda dastur kodining butun matni oldindan analiz qilinib, mashina tiliga to‘liq tarjima qilinadi. Natijada yakuniy bajariladigan (executable) fayl hosil bo‘ladi. Bu fayl foydalanuvchi tomonidan alohida ishlatilishi mumkin. Kompilyatsiya bosqichida sintaksis xatoliklari, ma’lumot turlari va boshqa muhim jihatlar tekshiriladi, bu esa dastur ishonchliligini oshiradi. Kompilatsiya qilingan dasturlar odatda tez ishlaydi, chunki bajarilish vaqtida qo‘shimcha tarjima qilishga ehtiyoj qolmaydi.

Interpretatsiya usulida esa kod qadam-baqadam tahlil qilinadi va darhol bajariladi. Har bir buyruq alohida o‘qilib, bajarilishi sababli dastur bajarilish tezligi kompilatsiyalangan dasturlarga nisbatan sekinroq bo‘ladi. Biroq interpretatorlar kodni tezda sinab ko‘rish va xatolarni aniqlashda qulay hisoblanadi. Bu yondashuv asosan ssenariy tillarida, masalan Python yoki JavaScript'da keng qo‘llaniladi.

Yuqoridagi ikki yondashuvdan tashqari, hozirgi kunda Just-In-Time (JIT) texnologiyasiga asoslangan aralash yondashuvlar ham mavjud. Unda dastur avval

ISSN 2181-3187

qisman kompilyatsiya qilinib, so‘ng bajarilish vaqtida zarur bo‘lgan qismlar darhol tarjima qilinadi. Bu esa kompilyatsiya va interpretatsiyaning afzalliklarini birlashtirgan usul bo‘lib, samaradorlik va moslashuvchanlikni ta’minlaydi.

Dastur bajarilishining so‘nggi bosqichi – bu kodni operatsion tizim yordamida qurilma resurslarida real ishga tushirishidir. Operativ xotira ajratilishi, protsessor resurslari bilan ishlash, kirish-chiqish tizimlari bilan o‘zaro aloqa kabi jihatlar aynan shu bosqichda amalga oshadi. Dastur natijasi foydalanuvchiga chiqariladi yoki boshqa tizimlarga uzatiladi.

Dastur bajarilishining muhim jihatlaridan biri bu — dastur qanday vosita orqali bajarilishi: **kompilyator** yoki **interpretator** orqali. Bu ikki yondashuv o‘ziga xos ishlash mexanizmlariga ega bo‘lib, har birining afzalliklari va cheklovlari mavjud. Kompilyator va interpretator dasturchi yozgan yuqori darajadagi kodni kompyuter tushunadigan mashina tiliga tarjima qilishga xizmat qiladi, ammo bu tarjima jarayoni va bajarilish vaqti jihatdan ular keskin farq qiladi.

Kompilyator dastur kodining butun matnini bir vaqtning o‘zida o‘qib, uni mashina tiliga to‘liq tarjima qiladi va natijada alohida bajariladigan (executable) fayl yaratadi. Bu fayl foydalanuvchi tomonidan to‘g‘ridan-to‘g‘ri ishga tushiriladi va bajarilishi uchun endi dastlabki kod kerak bo‘lmaydi. Kompilyatsiya jarayonida sintaksis xatolari aniqlanadi, bu esa dastur ishonchliligini oshiradi. Kompilyator asosida ishlovchi dasturlash tillariga **C**, **C++**, **Go** kabi tillarni misol qilib keltirish mumkin. Kompilyatsiya orqali ishlovchi dasturlar odatda tezroq ishlaydi, chunki ular bajarilish vaqtida hech qanday qo‘shimcha tarjimaga muhtoj emas.

Boshqa tomondan, **interpretator** dasturchi yozgan kodni qadam-baqadam o‘qiydi va darhol bajaradi. Ya’ni, har bir buyruq o‘qilgach, shu zahoti bajariladi. Interpretatsiya jarayoni kompilyatsiyaga qaraganda ko‘proq xotira va vaqt talab qilishi mumkin, chunki har bir ishga tushirishda kod qayta-qayta tahlil qilinadi. Shunga qaramay, interpretator orqali dastur tezda sinab ko‘rilishi va xatoliklar tezda topilishi mumkin. Bu yondashuv soddaroq va dinamik tillarda, masalan, **Python**, **JavaScript**,

ISSN 2181-3187

Ruby kabi tillarda keng qo‘llaniladi. Interpretatsiya dasturlarni ishlab chiqish bosqichida tezkor tahlil va test qilish uchun juda qulaydir.

Ba’zi zamonaviy tillar, masalan, **Java** yoki **C#**, ikkala yondashuvni o‘zida birlashtirgan: avval dastur **bayt-kodga** kompilyatsiya qilinadi, keyin esa **virtual mashina** orqali interpretatsiya yoki JIT (Just-In-Time) orqali bajariladi. Bu kombinatsiyalangan yondashuv samaradorlik va platformalararo ishlash imkonini beradi.

Umumiy olib qaraganda, kompilyator va interpretatorlarning har biri ma’lum vazifalar uchun qulaylik yaratadi. Dastur yaratishda qanday yondashuv tanlanishini loyiha talablari, ishlash tezligi, platforma mustaqilligi va dasturchining maqsadlari belgilaydi. Dastur bajarilishi mexanizmini chuqur tushunish esa samarali va sifatli dasturlar yaratishning muhim kalitidir.

Dastur bajarilish texnologiyalari yildan-yilga takomillashib, nafaqat samaradorlik, balki ko‘p platformalilik, xavfsizlik va ishlash tezligini oshirishga qaratilmoqda. An’anaviy kompilatsiya va interpretatsiya usullari bilan bir qatorda hozirgi kunda aralash yondashuvlar, virtual mashinalar, hamda Just-In-Time (JIT) kompilyatsiya kabi ilg‘or texnologiyalar keng qo‘llanilmoqda. Bu texnologiyalar dasturlarni samarali bajarish, resurslardan oqilona foydalanish va kodni xavfsiz muhitda ishga tushirish imkonini beradi.

Zamonaviy texnologiyalardan biri bu — **virtual mashinalar** (VM, Virtual Machines) bo‘lib, ular dastur bajariladigan muhit sifatida xizmat qiladi. Masalan, Java dasturlari **Java Virtual Machine (JVM)** orqali bajariladi. JVM dasturchi yozgan Java kodini avval **bayt-kod** ko‘rinishiga kompilyatsiya qiladi, so‘ng ushbu bayt-kodni JVM orqali interpretatsiya yoki JIT orqali mashina tiliga aylantirib bajaradi. Virtual mashina yondashuvi dasturga ko‘p platformalilik (platforma mustaqilligi) xususiyatini beradi, ya’ni bir martalik kompilyatsiyadan so‘ng dastur har qanday operatsion tizimda ishlay oladi — agar u yerda JVM mavjud bo‘lsa. Bu esa zamonaviy dasturlarda katta qulaylik yaratadi.

ISSN 2181-3187

Yana bir muhim texnologiya bu — **Just-In-Time (JIT) kompilyatsiya** hisoblanadi. JIT yondashuvi kompilyatsiya va interpretatsiya metodlarini birlashtiradi. Dastur bajarilishi vaqtida zarur bo‘lgan kod qismlari **real vaqt** rejimida kompilyatsiya qilinadi va shu zahoti bajariladi. Natijada dastur ishlashi tezlashadi, chunki JIT kompilator tez-tez ishlatiladigan kod segmentlarini mashina tiliga oldindan aylantirib, ularni qayta-qayta tahlil qilish zaruratini yo‘q qiladi. Bu yondashuv ko‘p hollarda .NET platformasi, Java va boshqa yuqori darajadagi tillarda qo‘llaniladi. Zamonaviy bajarilish muhitlari ko‘pincha xotira boshqaruvi (memory management), xatoliklarni avtomatik aniqlash, xavfsiz bajarilish muhiti kabi qo‘shimcha imkoniyatlarni ham taqdim etadi. Masalan, .NET va JVM platformalari avtomatik xotira tozalash (Garbage Collection) tizimiga ega bo‘lib, bu dastur samaradorligini oshiradi va xotira to‘kilishini oldini oladi.

Zamonaviy bajarilish texnologiyalarining afzalliklari nafaqat dasturchilar, balki foydalanuvchilar uchun ham muhimdir. Dastur tez ishlaydi, kamroq resurs talab qiladi, xavfsizlik me’yorlariga javob beradi va turli qurilmalarda barqaror ishlay oladi. Shu sababli zamonaviy dasturiy ta’minot yaratishda bajarilish texnologiyalarini to‘g‘ri tanlash katta ahamiyatga ega bo‘lib, bu tanlov dastur sifatini belgilovchi muhim omillardan biridir.

Dasturlarni bajarilishi — dasturlash va axborot texnologiyalari sohasida eng muhim va asosiy tushunchalardan biridir. Dasturchi yozgan kod qanday ishlashi, qaysi vositalar yordamida mashina tiliga aylantirilishi va qanday texnologiyalar asosida bajarilishi dastur samaradorligi, xavfsizligi va ko‘p platformalilik xususiyatlariga bevosita ta’sir qiladi. Ushbu maqolada dasturlarni bajarishning klassik usullari — kompilyatsiya va interpretatsiya, ularning o‘zaro farqlari, afzallik va kamchiliklari, shuningdek zamonaviy texnologiyalar – virtual mashinalar, JIT kompilyatsiya, va xotira boshqaruvi kabi masalalar ko‘rib chiqildi.

Bugungi kunda dasturlarni yaratish jarayonida faqatgina kod yozish emas, balki uni qanday bajarish, qaysi texnologiyaga asoslanishini puxta o‘rganish dasturchidan chuqur bilim va amaliy ko‘nikmalarni talab etadi. Zamonaviy bajarilish muhitlari

orqali dasturlarni tezkor, barqaror, resurslarga tejamkor va xavfsiz tarzda ishlab chiqish imkoniyatlari kengayib bormoqda. Ayniqsa, JVM va .NET kabi platformalar dasturiy ta’minotning turli tizimlarda mos va muammosiz ishlashini ta’minlaydi, bu esa zamonaviy ilovalarda muhim ahamiyat kasb etadi.

Xulosa qilib aytganda, dastur bajarilish mexanizmlarini chuqur o‘rganish va amaliyotda qo‘llay bilish zamonaviy dasturchilar uchun zarur kompetensiyalardan biridir. Bu nafaqat kod samaradorligini oshiradi, balki butun dastur arxitekturasining to‘g‘ri qurilishiga asos bo‘ladi. Kelajakda bu boradagi texnologiyalar yanada takomillashib boradi va bu esa dasturlash sohasining innovatsion rivojlanishiga kuchli turtki beradi.

Foydalanilgan adabiyotlar

1. Tanenbaum, A. S., & Bos, H. (2015). *Modern Operating Systems*. Pearson.
2. Aho, A. V., Lam, M. S., Sethi, R., & Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools*. Pearson.
3. Stallings, W. (2018). *Operating Systems: Internals and Design Principles*. Pearson.
4. Bryant, R. E., & O'Hallaron, D. R. (2015). *Computer Systems: A Programmer's Perspective*. Pearson.
5. Scott, M. L. (2005). *Programming Language Pragmatics*. Morgan Kaufmann.
6. Nisan, N., & Schocken, S. (2005). *The Elements of Computing Systems*. MIT Press.
7. Tojimamatov, I., & Adxamova, C. (2025). AMALIY TIZIMLARDA BERILGANLAR BAZASINI BOSHQARISH TIZIMLARI O ‘RNI. *Академические исследования в современной науке*, *4*(21), 77-82.
8. Tojimamatov, I., & Fazliddinov, X. (2025). BERILGANLAR BAZASI ADMINISTRATORI VA UNING XUSUSYATLAR. *Академические исследования в современной науке*, *4*(21), 90-95.

9. Tojimamatov, I. (2025). BERILGANLAR BAZASIDA AXBOROT ALMASHINISH SXEMASI HAMDA TURLARI. *Академические исследования в современной науке*, *4*(21), 71-76.

10. Tojimamatov, I., & Zaylobiddinova, M. (2025). AUDIO MA'LUMOTLAR TAQDIMOT. *TRANSFORMING EDUCATION THROUGH SCIENTIFIC DISCOVERY*, *1*(1), 38-44.

11. Tojimamatov, I., & Ismoilov, J. (2025). KRIPTOGRAFIK KALITLARNING KELAJAGI: XAVFSIZLIKNI TA'MINLASHDA YANGI YO 'NALISHLAR. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, *1*(1), 56-63.

12. Tojimamamtov, I. (2025). РАҚАМЛИ ИҚТИСОДИЁТНИНГ ИЖТИМОИЙ-ИҚТИСОДИЙ ЖАРАЁНЛАРНИ БОШҚАРИШ ВА РЕЖАЛАШДАГИ МОҲИЯТИ. *DIGITAL TRANSFORMATION AND ARTIFICIAL INTELLIGENCE*, *3*(1), 229-233.

13. Tojmamatov, I., Sharofutdinov, I., & Saydolimova, G. (2025). BILIMLAR OMBORI. BILIMLAR OMBORINING TUZILISHI. *EDUCATION AND RESEARCH IN THE ERA OF DIGITAL TRANSFORMATION*, *1*(1), 431-437.

14. Tojimamatov, I., & Metinboyeva, F. (2025). TESKARI MUNOSABATDA MUNOSABATLAR USTIDAGI AMALLAR. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, *1*(1), 202-214.

15. Tojimamatov, I., & Abdulhafizov, I. (2025). OBYEKTLAR VA ATRIBUTLAR. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, *1*(1), 107-112.

16. Tojimamatov, I., & Qirg'izboyev, D. (2025). SQL SO 'ROVLAR TILI STRUKTURASI. SQL-STRUKTURALANGAN SOROVLAR TILI. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, *1*(1), 64-71.

17. Tojimamatov, I., & Sahobiddinov, A. (2025). BERILGANLAR BAZISI USTIDA HISOBOTLARNI SHAKILLANTIRISH. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, *1*(1), 83-90.

18. Tojimamatov, I., & Islomova, T. (2025). MA'LUMOTLARNI IKKILIK SANOQ SISTEMASIDA KODLASH VA DEKODLASH. *BRIDGING THE GAP: EDUCATION AND SCIENCE FOR A SUSTAINABLE FUTURE*, *1*(1), 136-143.

19. Тоджимаматов И. Н. и Иминова Г. И. (2025). SEMANTIK OBEKT MODELI VA KATTA MA'LUMOTLAR (BIG DATA). *ОБРАЗОВАНИЕ И НАУКА В XXI ВЕКЕ*, (58-3).

20. Isroil, T. (2025). T-SQL Operatorlari: Ma'lumotlar Bazasi Operatsiyalarini Tezlashtirishning Asosiy Vositalari. *Modern education and development*, *18*(6), 31-44.

21. Nurmamatovich, T. I., & Sidiqjonovna, I. D. (2025). AUDIO MA'LUMOTLAR TAQDIMOT. *MODERN PROBLEMS IN EDUCATION AND THEIR SCIENTIFIC SOLUTIONS*, *1*(5), 125-129.

22. Isroil, T. (2025). BERILGANLAR BAZASIDA KONSEPTUAL MODEL YARATISH: NAZARIY ASOSLAR VA AMALIY YONDOSHUVLAR. *Modern education and development*, *18*(6), 51-63.

23. Tojimamatov, I., & Ahmataliyeva, S. (2025). BERILGANLARNI MARKAZLASHGAN TARZDA BOSHQARISH TAMOYILLARI. *Академические исследования в современной науке*, *4*(21), 59-64.

24. Tojimamatov, I., & Muslimaxon, N. (2025). AUDİO VA VİDEO KODLASH: RAQAMLİ TEXNALOGİYALAR DAVRİ. *YANGI O 'ZBEKISTON, YANGI TADQIQOTLAR JURNALI*, *2*(2), 159-165.