C++ DASTURLASH TILI BOSHQARUV OPERATORLARI: O’TISH OPERATORLARI

Abdullayev Shaxboz Solijon o‘g‘li
FarDU Axborot texnologiyalari kafedrasi katta o'qituvchisi
shaxbozfardu2023@gmail.com
ORCID ID 0000-0001-9382-732X
Davlatova Mohigul Ikromjon qizi.
Farg‘ona davlat universiteti talabasi
mohiguldavlatova4@gmail.com

Annotatsiya: *Ushbu maqolada C++ dasturlash tilidagi boshqaruv operatorlaridan biri bo‘lgan o‘tish operatorlari haqida batafsil ma’lumot berilgan. O‘tish operatorlari dastur oqimini aniq va samarali boshqarish imkonini beradi. Maqolada break, continue, goto, return kabi operatorlarning sintaksisi, ishlash mexanizmi va amaliy qo‘llanilishi misollar bilan yoritilgan. Har bir operatorning afzalliklari, cheklovlari va foydali foydalanish holatlari tahlil qilingan..*

Kalit so‘zlar: *C++, boshqaruv operatorlari, o‘tish operatorlari, break, continue, goto, return, dastur oqimi.*

Аннотация: *В данной статье подробно рассматриваются операторы перехода в языке программирования C++. Операторы перехода позволяют эффективно управлять потоком выполнения программы. Описаны синтаксис, механизм работы и практическое применение таких операторов, как break, continue, goto, return. Также проанализированы преимущества, ограничения и рекомендуемые случаи их использования.*

Ключевые слова: *C++, операторы управления, операторы перехода, break, continue, goto, return, поток выполнения.*

Abstract: *This article provides a detailed overview of control flow transfer statements in the C++ programming language. Transfer statements allow programmers to manage the flow of execution precisely. The paper explains the syntax, working principles, and practical usage of operators like break, continue, goto, and return, along with their advantages, limitations, and recommended use cases.*

Keywords: *C++, control statements, transfer operators, break, continue, goto, return, program flow.*

Zamonaviy dasturlash tillarida dastur oqimini boshqarish — ya’ni buyruqlar qanday ketma-ketlikda bajarilishini belgilash — muhim va zaruriy masalalardan biridir. C++ dasturlash tili bu borada keng imkoniyatlarga ega bo‘lib, turli xil boshqaruv operatorlari orqali foydalanuvchiga dastur strukturasini aniq nazorat qilishga yordam beradi. Ular orasida o‘tish operatorlari muhim o‘rin tutadi.

O‘tish operatorlari — bu buyruqlarning bajarilish tartibini odatdagidan farqli ravishda o‘zgartirish imkonini beruvchi vositalardir. Ular yordamida sikllardan istalgan vaqtda chiqish, ma’lum shart asosida keyingi iteratsiyaga o‘tish yoki hatto kodning boshqa nuqtasiga sakrash mumkin bo‘ladi. Bunday operatorlar dastur soddaligini ta’minlashi, ba’zan esa noto‘g‘ri qo‘llanilganda kodni murakkablashtirishi mumkin.

Mazkur maqolada C++ tilida mavjud bo‘lgan break, continue, goto, va return operatorlarining ishlash mexanizmi, ularning dasturdagi roli va ulardan qanday to‘g‘ri foydalanish mumkinligi haqida batafsil so‘z yuritiladi. Har bir operator misollar yordamida ko‘rib chiqiladi va ularning qachon va qanday holatda ishlatilishi tavsiya etilishi tahlil qilinadi.

C++ dasturlash tilida mavjud bo‘lgan o‘tish operatorlari dasturchiga dasturning bajarilish tartibini yanada moslashtirish va nazorat qilish imkonini beradi. Bunday operatorlar yordamida ma’lum bir shart bajarilganda sikldan chiqish, navbatdagi iteratsiyaga o‘tish, yoki hatto kodning boshqa qismiga sakrash mumkin bo‘ladi. Odatda, o‘tish operatorlari to‘g‘ridan-to‘g‘ri kod oqimini o‘zgartiruvchi buyruqlar hisoblanadi. Ularning noto‘g‘ri ishlatilishi kodni murakkablashtirishi mumkin bo‘lsa-da, to‘g‘ri yondashilganda ular dasturchi ishini sezilarli darajada yengillashtiradi.

Break operatori dastur bajarilishi davomida ma’lum bir shart bajarilganda sikldan yoki switch konstruksiyasidan chiqib ketishni ta’minlaydi. Dasturda ushbu operator ko‘pincha foydalanuvchi kiritgan qiymatlar asosida siklni to‘xtatish zarur bo‘lgan holatlarda qo‘llaniladi. Masalan, foydalanuvchi musbat son kiritmaguncha sikl ishlaydi, ammo kerakli qiymat kiritilgach, break operatori yordamida sikl tugatiladi. Bu dastur samaradorligini oshiradi va keraksiz ishlov berishlardan saqlaydi. Bundan tashqari, break operatori switch tanlash operatorida ham juda keng

qo‘llaniladi. Har bir case dan keyin break yozilishi natijasida boshqa case holatlariga avtomatik o‘tishning oldi olinadi. Bu esa dasturning lo‘nda va aniq ishlashiga xizmat qiladi.

Continue operatori esa, aksincha, sikldan chiqmaydi, balki faqatgina hozirgi iteratsiyani tashlab ketib, keyingi aylanishga o‘tadi. Bu operator odatda istalmagan shartlar yuzaga kelganda keyingi qadamga o‘tish kerak bo‘lgan holatlarda qo‘llaniladi. Masalan, foydalanuvchi manfiy son kiritgan bo‘lsa, bu qiymat ustida hisob-kitob ishlari bajarilmaydi va sikl davom ettiriladi. Bunday yondashuv dasturda keraksiz operatsiyalarni kamaytiradi va ma’lumotlar ustida faqat zarur amallar bajarilishini ta’minlaydi. Continue operatoridan to‘g‘ri foydalanish kodni soddalashtirish bilan birga, uning o‘qilishi va tushunilishini ham yengillashtiradi.

Goto operatori esa dasturda istalgan nuqtaga sakrashni amalga oshiradi. U yordamida dasturning bajarilishini biror belgilangan joyga olib borish mumkin. Bunda operatorga tegishli etiketka qo‘yiladi va dastur shu belgiga sakrash orqali kerakli kodni bajaradi. Ammo bu operator C++ dasturlash tilida juda ehtiyotkorlik bilan ishlatiladi, chunki noto‘g‘ri ishlatilganda dastur oqimi tartibsiz holga kelib qolishi mumkin. Dasturchilar bunday operatorni faqat juda zarur va boshqa yechim mavjud bo‘lmagan holatlargina qo‘llashlari tavsiya etiladi. Masalan, murakkab menyu tizimlari yoki foydalanuvchi noto‘g‘ri amal bajarganda dasturning boshiga qaytarilishi zarur bo‘lganda goto operatori foydali bo‘lishi mumkin. Biroq ko‘plab zamonaviy yondashuvlar bunday holatlarda sikl va funksiyalar yordamida yechim topishni afzal ko‘radi.

Return operatori C++ dasturlash tilidagi eng muhim o‘tish operatorlaridan biridir. Uning yordamida funksiyadan chiqish va kerakli natijani chaqiruvchi kodga qaytarish mumkin bo‘ladi. Agar funksiya hech qanday qiymat qaytarmasa, ya’ni void turida bo‘lsa, unda return operatori shunchaki funksiyani tugatish uchun ishlatiladi. Bu operator dastur strukturasini aniq tashkil etish, funksiya vazifasini to‘liq bajarish va kerakli qiymatlarni kerakli joyga uzatish imkonini beradi. Funksiyalarni modulga bo‘lishda return operatorining roli ayniqsa muhim hisoblanadi. Return yordamida dasturchi funksiyaning yakuniy natijasini istalgan shaklda boshqarishi mumkin.

Umuman olganda, o‘tish operatorlari dasturlash jarayonida kod oqimini to‘liq nazorat qilish imkonini beruvchi vositalardir. Ulardan to‘g‘ri foydalanish dasturni lo‘nda, aniq va samarali qiladi. Bunday operatorlar yordamida murakkab shartli

vaziyatlarni oddiy kod bloklari bilan boshqarish mumkin. Shu bilan birga, har bir operatorni maqsadga muvofiq qo‘llash muhim. Masalan, break operatori siklni to‘xtatishda, continue esa iteratsiyani o‘tkazib yuborishda foydalidir. Goto operatori faqat zarur holatlarda, return esa funksiyalarning lo‘nda va to‘g‘ri ishlashini ta’minlashda qo‘llanilishi lozim.

Dasturchi sifatida bu operatorlarning har biri qanday ishlashini chuqur o‘rganish, ularni qachon ishlatish kerakligini anglash va noto‘g‘ri foydalanishdan saqlanish — sifatli dastur tuzishning ajralmas qismidir. Ayniqsa, dastur loyihalashtirilayotgan vaqtda, turli algoritmlarni rejalashtirishda o‘tish operatorlari yordamida dastur tuzilmasini soddalashtirish va natijada yaxshi natijaga erishish mumkin. Zamonaviy dasturlashda kodning o‘qilishi, parvarishi va kengaytirilishi muhim hisoblanganligi bois, o‘tish operatorlarini ehtiyotkorlik bilan qo‘llash va kod strukturaviy jihatdan silliq va tushunarli bo‘lishiga e’tibor berish tavsiya etiladi.

Xulosa

C++ dasturlash tilidagi o‘tish operatorlari yordamida dasturning bajarilish tartibini boshqarish, kerakli joyda siklni to‘xtatish, ayrim iteratsiyalarni tashlab ketish, funksiyadan chiqib ketish yoki dasturning boshqa nuqtasiga o‘tish mumkin bo‘ladi. Bu operatorlar dastur oqimini yanada moslashuvchan qilishga xizmat qiladi. Break va continue operatorlari odatda takrorlanuvchi jarayonlarda qulaylik yaratadi, return operatori funksiyalarni aniq va lo‘nda boshqarishga yordam beradi, goto esa maxsus holatlarda soddalashtirish uchun ishlatiladi. Biroq ularni o‘z o‘rnida, ehtiyotkorlik bilan qo‘llash lozim. Zero, noto‘g‘ri yondashuv kodni chalkashtirib yuborishi, tushunishni va texnik xizmat ko‘rsatishni qiyinlashtirishi mumkin. Shuning uchun bu operatorlarni chuqur tushunish, ularning har birini o‘z joyida ishlata bilish — har bir dasturchining asosiy vazifasidir. Mazkur mavzuni o‘rganish orqali o‘quvchi dastur tuzishda yanada samarador, aniq va tartibli yondashuvni qo‘llash imkoniyatiga ega bo‘ladi. Shuningdek, maqolada algoritmlarning amaliy dasturlashdagi o‘rni va ular bilan ishlashda e’tibor berilishi lozim bo‘lgan jihatlar haqida fikr yuritildi. Kelgusida ushbu mavzuni yanada chuqurlashtirib, algoritmlarni vizual ko‘rinishda taqqoslash, ularni grafik usullar bilan tahlil qilish orqali o‘rganishni kengaytirish mumkin.

Adabiyotlar ro‘yxati

1. Cormen, T.H., Leiserson, C.E., Rivest, R.L., & Stein, C. (2009). *Introduction to Algorithms*. MIT Press.
2. Knuth, D.E. (1998). *The Art of Computer Programming: Sorting and Searching*. Addison-Wesley.
3. Malik, D.S. (2010). *C++ Programming: From Problem Analysis to Program Design*. Cengage Learning.
4. Lafore, R. (2002). *Data Structures and Algorithms in C++*. Sams Publishing.
5. Sedgewick, R., & Wayne, K. (2011). *Algorithms*. Addison-Wesley.
6. Назаров, Р.Ю. (2020). *Алгоритмы и структуры данных*. Москва: Академия.
7. Алиев, Ш.Ш. (2021). *Zamonaviy algoritmlar nazariyasi*. Toshkent: Fan va texnologiya.
8. OnlineGDB. (2024). C++ Examples [https://www.onlinegdb.com]
9. GeeksForGeeks. (2024). Sorting Algorithms Explained [https://www.geeksforgeeks.org]
10. OpenGenus. (2024). Merge Sort vs Quick Sort.