STL KUTUBXONALARI

Toshkent Davlat Iqtisodiyot Universiteti
Andijon Fakulteti 2- bosqich AATT yo'nalish
70/23 guruh talabalari **Umarjanov Shoxrux**
Choriyev Mahmudjon
Ilmiy rahbar **Ismailov Alisher**

Annotatsiya: Dasturlash – kompyuterlar va boshqa mikroprotsessorli elektron mashinalar uchun dasturlar tuzish, sinash va o'zgartirish jarayonidan iborat. Odatda dasturlash yuqori savilayi dasturlash tillari (PHP, Java, C++, Pyhton) vositasida amalga oshiriladi. Bu dasturlash tillarining semantikasi odam tiliga yaqinligi tufayli dastur tuzish jarayoni ancha oson kechadi. Dasturlash jarayoni, odatda, quyidagi bosqichlarga bo'linadi: masalaning qo'yilishi; masalaning algoritmik tavsifini tuzish; masalani yuqori darajadagi programma tilida Dasturlash: masalani mashina tilida Dasturlash

Dasturlashning asosiy qismlaridan biri STL (Standart shablon kutubxonalari) biz bu mavzuni C++ dasturlash tilida yoritib beramiz. C++ Standart Shablon Kutubxonasi (STL) dasturchilarga ma'lumotlar strukturalari va algoritmlarni samarali boshqarish uchun umumiy yondashuvni taqdim etadi. STL ning asosiy qismi konteynerlar bo'lib, ular ma'lumotlarni saqlash va boshqarish uchun mo'ljallangan sinflar to'plamini o'z ichiga oladi. Konteynerlar odatda turlar va ma'lumotlar ketma-ketligini saqlash uchun ishlatiladi va ular turli xil ehtiyojlarga mos keladigan ko'rinishlarga ega.

C++ da STL (Standart Shablon Kutubxonasi) — bu shablonlar texnologiyasiga asoslangan kutubxona bo'lib, umumiy ma'lumotlar strukturalari va algoritmlarni o'z ichiga oladi. STL uch asosiy komponentdan iborat:

1. Konteynerlar (Containers): Ma'lumotlarni saqlash va boshqarish uchun maxsus sinflar to'plami.
2. Algoritmlar (Algorithms): Ma'lumotlar ustida bajariladigan umumiy amallar (masalan, `sort()`, `find()`.
3. Iteratorlar (Iterators): Konteyner elementlariga kirish va ularni boshqarish uchun foydalaniladigan mexanizmlar.

STLning asosiy maqsadi — kodni qayta foydalanish, dasturiy ta'minotning moslashuvchanligini ta'minlash va ma'lumotlar ustida samarali operatsiyalarni amalga oshirishdir. STL kutubxonasi konteyner sinflarini taqdim etadi, ular

yordamida ma'lumotlar tuzilmalarini yaratish va boshqarish imkoniyatiga ega bo‘lamiz.

Kalit so’zlar: STL, C++, Visual Studio, Code::Blocks, Iterator, Algaritm, Konteyner

Abstract: Programming is the process of designing, testing, and executing programs for computers and other microprocessor-based electronic machines. programming using high-level programming languages (PHP, Java, C++, Python). Since the semantics of these programming languages are close to humans, the programming process is much easier. The programming process is divided into stages: setting the problem; algorithmic description of the problem; Programming the problem in the above programming language: Programming the problem in machine language One of the programming works STL (Sdart Template Libraries) we cover this topic in C++ programming language. The C++ Standard Template Library (STL) provides programmers with common objects for manipulating data structures and algorithms. The food part of the STL are containers, which contain a collection of classes designed to store and control data. Containers are for storing sequences and have views that correspond to different types of data. In C++, the STL (Standard Template Library) is a free template-powered library that contains common data structures and algorithms. STL consists of three structural components:

1. Containers: A set of special classes for storing and managing data.
2. Algorithms (Algorithms): General operations performed on data (for example `sort()`, `find()`.
3. Iterators (Iterators): Recursive mechanisms for accessing and controlling container checks.

The goal of code processing is to process the code, provide it, and work on the data to perform STL production. The STL library provides classes with which we can get the container data structures we need and want.

Keywords: STL, C++, Visual Studio, Code::Blocks, Iterator, Algorithm, Container

Kirish: STL 1970-yillarning oxiri va 1980-yillarning boshlarida Aleksandr Stepanov tomonidan ishlab chiqilgan. C++ da umumiy dasturlashdan foydalanish usulini yaratishga intilgan.

Dastlabki maqsad algoritmlarga turli xil ma'lumotlar tuzilmalarida ularni qayta yozmasdan ishlash imkonini beradigan kutubxona yaratish edi. STL bir nechta asosiy tushunchalarni, jumladan, iteratorlar, konteynerlar va algoritmlarni taqdim etdi. Iteratorlar konteynerlardagi elementlarga kirishning yagona usulini taqdim etdi,

konteynerlarda esa vektorlar, ro'yxatlar va xaritalar kabi ma'lumotlar tuzilmalari mavjud. C++ tomonidan qabul qilish: 1990 yilda STL C++ hamjamiyatiga taqdim etildi va C++ ishlab chiquvchilari orasida mashhurlikka erisha boshladi. 1998 yilda C++ standartining birinchi nashri chiqarilishi bilan STL rasman C++ standart kutubxonasining bir qismi sifatida kiritildi.Evolyutsiya: STL keyingi C++ standartlarida (C++11, C++14, C++17, C++20 va boshqalar) takomillashtirildi va kengaytirildi, aqlli ko'rsatkichlar, parallellikni qo'llab-quvvatlash va diapazonga asoslangan xususiyatlarni joriy etdi. STL ning ta'siri STL kodni qayta ishlatish va samaradorlikni rag'batlantirish orqali C++ dasturlashiga sezilarli ta'sir ko'rsatdi. Uning umumiy dasturlash paradigmasi ishlab chiquvchilarga mavhum va moslashuvchan kod yozish imkonini beradi. U akademik va sanoat dasturlarida keng qo'llaniladigan zamonaviy C++ dasturlashning asosiy qismiga aylandi. C++ tilidagi standart andozalar kutubxonasi (STL) bir nechta muhim maqsadlarga xizmat qiladi: STL ning vazifalari umumiy dasturlashni ta’minlaydi,bu algaritmlarga kodni qayta yozmasdan turli xil ma’lumotlar turlarida ishlashga imkon beradi. Bu esa kodni qayta ishlatish va moslashuvchanlikni ta’minlaydi.

Yoritilgan mavzu: Standart andoza kutubxonalari (STL) C++ dasturlash tilida muhim bir bo’lagi hissoblanadi u C++ hamjamiyatiga katta katta hissa qo’shdi.Uning e’tiborga loyiq yutuqlarini sanab o’tadigan bo’lsak:

1) STL C++ tilida umumiy dasturlash tushunchasini joriy etgan birinchi kutubxonalardan biri edi. Bu ishlab chiquvchilarga har qanday ma'lumotlar turi bilan ishlaydigan algoritmlarni yaratishga imkon berdi, bu kodning qayta ishlatilishi va moslashuvchanligini sezilarli darajada oshirdi.

2) STL C++ 98 standartida standartlashtirilib, uni C++ tilining asosiy qismiga aylantirdi. Ushbu standartlashtirish turli kompilyatorlar va platformalarda izchil interfeys va xatti-harakatni ta'minlab, portativlik va ishonchlilikni oshirdi.

3) STL turli xil kompanentlar majmuasini taqdim etdi. Jumladan yuqorida ham tilga olib o’tdik:

Konteynerlar: (masalan, vectorlar, royxatlar, toplamlar, xotiralar) ma’lumotlarni saqlash uchun.

Algaritmlar: (masalan, saralash, qidirish) ma’lumotlarni manipulyatsiya qilish.

Iteratorlar: konteyner elementlarini samarali o’tkazish uchun.

Ushbu kompanentlar yordamida kutubxonalardan foydalanib kodlarni oson va vaqt sarlamasdan yozish imkoniyatini beradi.

4) STL yuqori unumdorlik uchun mo'ljallangan. Uning ko'pgina algoritmlari tezlik va samaradorlik uchun optimallashtirilgan bo'lib, ko'pincha foydalanish

qulayligini yo'qotmasdan past darajadagi dasturlash strategiyalaridan foydalanadigan usullardan foydalanadi. Bu STL ni yuqori unumli ilovalar uchun mos qiladi.

5) Yaratilganidan beri STL turli C++ standartlari orqali rivojlandi. C++ 11 da harakatlanuvchi semantika va lambda ifodalari, shuningdek C++ 17 da parallel algoritmlarni joriy etish kabi yaxshilanishlar STL ni zamonaviy dasturlashda tegishli va kuchli bo'lib qoldi.

6) STL ma'lumotlar tuzilmalari va algoritmlarni o'rgatishda hal qiluvchi rol o'ynadi. Uning aniq tatbiq etilishi va keng ko'lamli hujjatlari uni talabalar va o'qituvchilar uchun murakkab dasturlash tushunchalarini tushunishga yordam beruvchi manbaga aylantirdi.

Xar bir yutuqning o’ziga yarasha kamchiliklari bor bu esa STL ni ham chetlab o’tmadi uning ham o’ziga yarasha kamchiliklari mavjud, men hozir ma’lum miqdorda kamchiliklarini ham sanab o’tishga harakat qilaman. STL ning asosiy kamchiliklari :

1) Tik o'rganish egri chizig'i: STL foydalanuvchilar uchun qulay bo'lsa-da, uning barcha xususiyatlarini va ulardan qanday samarali foydalanishni to'liq tushunish vaqt talab qilishi mumkin, ayniqsa shablonlar va iteratorlar tushunchalarini qiyin deb topishi mumkin bo'lgan yangi boshlanuvchilar uchun.

2) Ishlash uchun qo'shimcha xarajatlar: STL ning umumiy dasturlash yondashuvi ishlash uchun jarimalarni kiritishi mumkin. Masalan, assotsiativ konteynerlarda (masalan, xaritalar) operatsiyalar uchun iteratorlardan foydalanish ushbu konteynerlar uchun mo'ljallangan a'zo funktsiyalaridan to'g'ridan-to'g'ri foydalanishga qaraganda sekinroq bo'lishi mumkin.

3) Cheklangan sozlanishi: Ba'zi ishlab chiquvchilar STL konteynerlarining standart ilovalarini sozlash imkoniyatlari nuqtai nazaridan cheklovchi deb hisoblashlari mumkin. Bu oldindan taqdim etilmagan o'ziga xos funksiyalarga muhtoj bo'lganlar uchun kamchilik bo'lishi mumkin.

4) Abstraktsiyaning umumiy xarajatlari: STL tomonidan taqdim etilgan abstraktsiyalar ba'zan samarasizlikka olib kelishi mumkin, ayniqsa, ushbu abstraktsiyalarni boshqarish uchun qo'shimcha xarajatlar ular taqdim etadigan foydadan ustun bo'lgan hollarda

5) Nosozliklarni tuzatishning murakkabligi: STL-dan foydalanadigan disk raskadrovka kodi, ayniqsa, shablon xatolari yoki iteratorni bekor qilish bilan bog'liq muammolarni hal qilishda murakkab bo'lishi mumkin. STL konteynerlari va

algoritmlarining asosiy mexanikasini tushunish bunday kodni samarali tuzatish uchun juda muhimdir.

Ushbu kamchiliklarga qaramasdan, ko'plab ishlab chiquvchilar STL dan foydalanishning afzalliklari, masalan, unumdorlikni oshirish va yaxshi sinovdan o'tgan komponentlardan foydalanish qobiliyati ko'pincha kamchiliklardan ustun ekanligini aniqlaydilar.

STL (standart andoza kutubxonalari) bu qanday dasturlarda ishlashini ko’ramiz:

Standart andozalar kutubxonasi (STL) dasturlash tili sifatida C++ dan foydalanadigan har qanday dastur yoki dasturda ishlaydi. STL C++ standart kutubxonasining bir qismi bo'lgani uchun u C++ ni qo'llab-quvvatlaydigan turli ishlab chiqish muhitlari va operatsion tizimlar bilan mos keladi. STL ma’lumotlar tuzilmalari, algoritmlar va xotirani samaraliroq boshqarish uchun C++ da ishlab chiqilgan ish stoli ilovalarida keng qo‘llaniladi. Visual Studio, Code::Blocks va CLion kabi mashhur integratsiyalashgan ishlab chiqish muhitlari (IDE) STLni qo'llab-quvvatlaydi. C++ ko'pincha o'rnatilgan tizimlarda ishlash uchun muhim ilovalar uchun ishlatiladi. STL ning yengil konteynerlari (masalan, "vektor" va "massiv") xotira va protsessordan foydalanish asosiy muammolar bo'lgan bunday muhitlarda foydalidir. Umumiy olib qaraydigan bo’lsak, STL deyarli har qanday C++ dasturida, ish stoli va mobil ilovalardan tortib, o'rnatilgan tizimlar, o'yin dvigatellari, ilmiy hisoblashlar va boshqalar uchun ishlatilishi mumkin, agar muhit C++ ni qo'llab-quvvatlasa. STL ning qachon ishlatilishi maqsadga muvofiq bo’lar edi: STL (Standart andozalar kutubxonasi) ko'pgina C++ dasturlash stsenariylarida juda foydali, lekin u, ayniqsa, umumiy ma'lumotlar tuzilmalari va algoritmlari bilan ishlashda foydalidir. STL turli xil foydalanish holatlari uchun optimallashtirilgan “vektor”, “ro‘yxat”, “xarita”, “to‘plam” va “deque” kabi konteynerlarni taklif etadi. Agar sizning dasturingiz ushbu ma'lumotlar tuzilmalaridan birini talab qilsa va siz ularni noldan amalga oshirishdan qochmoqchi bo'lsangiz, STL juda mos keladi. Misollar ma'lumotlar to'plamini boshqarish, kalit-qiymat xaritalarini yaratish yoki dinamik massivlarni amalga oshirishni o'z ichiga oladi. STL ishlash va xotirani boshqarish uchun yaxshi optimallashtirilgan. Agar sizga tezkor saralash (`std::sort`), qidirish (`std::find`, `std::binary_search`) yoki katta ma`lumotlar to`plamlarini o`tkazish kerak bo`lsa, STL samarali va ishlatish uchun qulay algoritmlarni taqdim etadi. - vaqt ichida murakkablik kafolatlari. STL C++ standart kutubxonasining bir qismi bo‘lib, turli kompilyatorlar va platformalarda qo‘llab-quvvatlanadi, bu esa uni platformalararo ilovalarni ishlab chiqish uchun ideal qiladi. Turli muhitlarda izchil xatti-harakatlar va interfeysni ta'minlaydi.

STL eng yaxshi tanlov bo'lmasligi mumkin, agar:
- Siz o'zingizning foydalanish holatlaringizga xos xotira yoki ishlashni optimallashtirish ustidan juda nozik nazoratga muhtojsiz.
- STL osonlikcha sig'dira olmaydigan maxsus ma'lumotlar tuzilmalari yoki algoritmlari talab qilinadi.

C++ tilidagi standart andozalar kutubxonasi (STL) birinchi navbatda ishlab chiquvchilar uchun qimmatli manba hisoblanadi, chunki u ma’lumotlar tuzilmalari va algoritmlarini samarali boshqarish va boshqarish uchun yaxshi tashkil etilgan vositalar to‘plamini taqdim etadi. STL dan qachon foydalanish kerak ? Keling shu haqida ko’rib chiqamiz.

Qachon samaradorlik kaliti: STL vektorlar, ro‘yxatlar va xaritalar kabi optimallashtirilgan, oldindan tuzilgan ma’lumotlar tuzilmalarini taqdim etadi, bu esa noldan murakkab ma’lumotlarni qayta ishlashni qo‘lda kodlash bilan solishtirganda vaqtni tejaydi.

Tez prototiplash va ishlab chiqish: STL yordamida ishlab chiquvchilar ushbu funksiyalarni mustaqil ravishda amalga oshirish zaruratisiz standart ma’lumotlar bilan ishlashni (masalan, saralash yoki qidirish) tezda sozlashlari mumkin, bu esa uni tez sur’atda rivojlanayotgan muhitlarda foydali qiladi.

Kod ishonchliligi va texnik xizmat ko‘rsatish: STL funksiyalari yaxshi sinovdan o‘tgan va keng tarqalgan bo‘lib foydalaniladi, bu esa xatolar ehtimolini kamaytiradi va kod barqarorligini oshiradi. Bu, ayniqsa, kod sifatini saqlash muhim bo'lgan loyihalar uchun foydalidir.

Qayta foydalanish mumkin va kengaytiriladigan kod: STL konteynerlari va algoritmlari masshtabni boshqarish uchun yaratilgan bo‘lib, ularni ma’lumotlarga ehtiyoj sezilarli darajada farq qiladigan ilovalar uchun mos qiladi.

STL kim uchun muhim: dasturchi yoki foydalanuvchi?

Albatta ishlab chiquvchilar uchun: STL birinchi navbatda ishlab chiquvchilar uchun zarur, chunki u vaqtni tejaydi, kod murakkabligini kamaytiradi va kodlar bazasini toza va boshqariladigan qiladi. Bu ishlab chiquvchilarga standart operatsiyalar uchun g'ildirakni qayta ixtiro qilmasdan turib, o'z loyihasining o'ziga xos jihatlariga e'tibor qaratish imkonini beradi. Foydalanuvchilar uchun: Garchi foydalanuvchilar STL bilan bevosita o'zaro aloqada bo'lmasalar ham, ular bilvosita foyda ko'radilar. STL-dan foydalanadigan kod ko'pincha tezroq va ishonchliroq bo'lib, kamroq xatoliklarga ega bo'lgan yaxshi ishlaydigan ilovalarga olib keladi. Masalan, tezkor ma'lumotlarni olish uchun STL-dan foydalanadigan ilovalar tezroq javob vaqtlariga ega bo'lishi mumkin, bu esa foydalanuvchi tajribasini oshiradi. Xulosa qilib

aytadigan bo’lsak, STL ishlab chiquvchilar uchun juda muhim, ammo dasturiy ta'minotda sifatni yaxshilash orqali foydalanuvchilarga bilvosita ustunlik beradi. Bu samarali, ishonchli va qo'llab-quvvatlanadigan kodni yaratmoqchi bo'lgan har bir kishi uchun C++ tilida amaliy tanlovdir.

XULOSA

Standart andozalar kutubxonasi (STL) C++ shablon sinflari va funksiyalarining kuchli to‘plami bo‘lib, ma’lumotlarni boshqarish va algoritmlarni bajarish uchun samarali, qayta foydalanish mumkin bo‘lgan komponentlarni taqdim etadi. Ushbu maqolada biz konteynerlar,iteratorlar va algaritmlar ular bilan ishlashning muhim tomonlarini, C++ dasturlash tilida STL kutubxonasi orqali konteynerlar bilan qanday ishlashni ko’rib chiqdik. Konteynerlar yordamida dasturlash jarayonida ma’lumotlarni saqlash va ularga ishlov berish sezilarli darajada soddalashadi va samaradorligi oshadi. Ma’lumotlarning har xil tuzilmalari bilan ishlashda mos keladigan konteynerlar tanlanadi, bu esa ma’lumotlarga tezkor va samarali kirishni ta’minlaydi. C++ dasturlash tilida konteynerlar va ularga mos algoritmlar yordamida murakkab ma’lumotlarni samarali boshqarish imkoniyati yaratiladi. STL kutubxonasi dasturchilarga nafaqat konteynerlarni, balki ular ustida ishlaydigan algoritmlarni ham taqdim etadi, bu esa kodni qisqartirish va optimallashtirish uchun qulay imkoniyatdir. Konteynerlar va ularning algoritmlari dasturiy ta’minotni ishlab chiqish jarayonida dasturchilarga ko’plab murakkab vazifalarni hal qilishda yordam beradi.

Ushbu maqoladi STL ning hayotimizda qanchalikm darajada muhim ahamiyatga ega ekanligi u kim uchun muhim, qanday yollarda foydalanish mumkinligi haqida yoritib berilgan. Maqolada STL kim uchun muhim foydalanuvchi uchunmi yoki dasturchi uchunmi bularning hammasi yotirib o’tilgan. STLning ichida foydalanishimiz mumkin bo’lgan bir qancha massivlar haqida ham yoritilgan. Umuman olganda STL ning hayotimizda muhim ahamiyatga ega ekanligini korishimiz mumkin. STL dasturnyozishda bir qancha vaqtni tejash imkonini ham beradi. Dasturchilar uchun STL bilan ishlashning asosiy tamoyillarini bilish va ularni turli amaliyotlarda qo’llay olish muhimdir. Umuman olganda maqolada STL haqida to’liq ma’lumot olish mumkin.

FOYDALANILGAN ADABIYOTLAR:

[1] Abdurakhmonova, N., Alisher, I., & Toirova, G. (2022, September). Applying Web Crawler Technologies for Compiling Parallel Corpora as one Stage of Natural Language Processing. In *2022 7th International Conference on Computer Science and Engineering (UBMK)* (pp. 73-75). IEEE.

[2] Abdurakhmonova, N., Alisher, I., & Sayfulleyeva, R. (2022, September). MorphUz: Morphological Analyzer for the Uzbek Language. In *2022 7th International Conference on Computer Science and Engineering (UBMK)* (pp. 61-66). IEEE.

[3] NAZIROVA, E., ABDURAKHMONOVA, N., & ALISHER, I. Exploring Linguistic Roots (stem) and Word Categories in Uzbek Language through Advanced Natural Language Processing Techniques for Text Analysis.

[4] Ismailov, A. S., Akbarov, A., Qodirova, G. T. X. Q., & Yigitaliyeva, M. (2023). Dasturiy ta'minotni ishlab chiqish bosqichlari. *Science and Education*, *4*(3), 187-191.