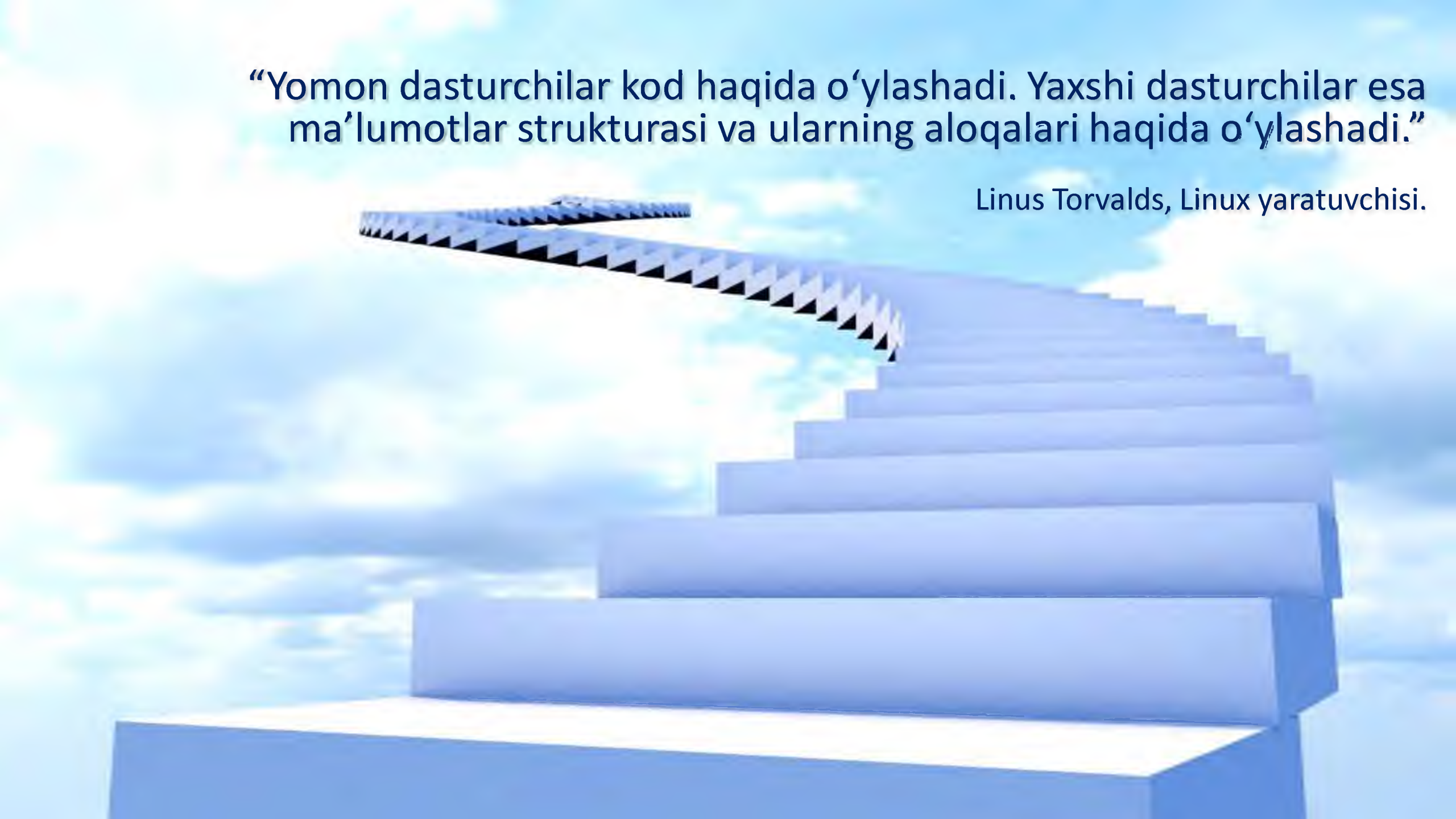
Ma’lumotlarning abstrak turlari va ma’lumotlar strukturalari. Stek, Navbat, Vektor, Ro‘yxat

2 - Ma’ruza

“Yomon dasturchilar kod haqida o‘ylashadi. Yaxshi dasturchilar esa ma’lumotlar strukturasi va ularning aloqalari haqida o‘ylashadi.”

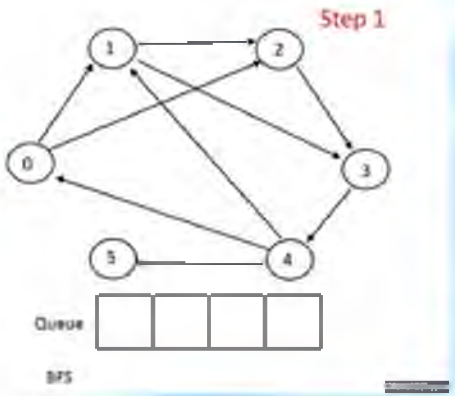
Linus Torvalds, Linux yaratuvchisi.



MA’RUZA REJASI

1. Ma’lumotlar strukturasi
2. Abstrakt ma’lumotlar turi
3. Stek
4. Navbat
5. Vektor
6. Ro‘yxat

Ma’lumotlar strukturasi



- **Ma’lumotlar strukturasi** (ing. data structure) - bu hisoblashda turli xil bir tipli va (yoki) mantiqiy bog‘liq ma’lumotlarni saqlash va qayta ishlashga imkon beradigan dastur birligi. Ma’lumotlarni qo‘shish, izlash, o‘zgartirish va yo‘q qilish uchun ma’lumotlar tarkibi uning interfeysini tashkil etadigan funksiyalar to‘plamini taqdim etadi.

Ma’lumotlar strukturasi

“Ma’lumotlar strukturasi” atamasining bir-biriga yaqin bo‘lgan bir nechta ma’nolarini anglatuvchi variantlari mavjud:			
Ma’lumotlarning abstrakt turi	Ma’lumotlarning ba’zi bir abstrakt turlarini realizatsiya qilish	Ma’lumotlar tipining nusxasi, masalan, aniq bir **ro‘yxat**	Funksional dasturlash kontekstida o‘zgarishlarda davom etadigan noyob identifikator

Funksional va imperativ dasturlashda ma’lumotlar strukturalarini taqqoslash.

Kamida ikkita sababga ko‘ra funksional tillar uchun ma’lumotlar tuzilmalarini loyihalash imperativ tillarga qaraganda ancha qiyin:

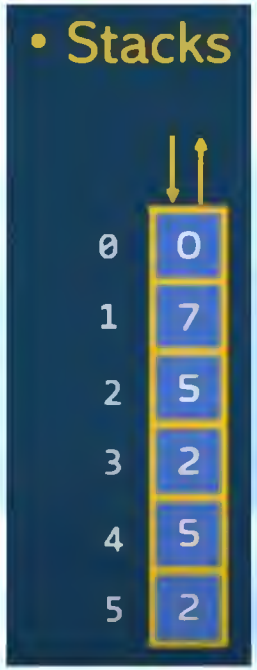
1. Deyarli barcha ma’ lumotlar strukturalari aniq funksional uslubda ishlatilmaydigan o‘zlashtirishlardan og‘ir foydalanadilar;
2. Ma’ lumotlarning funksional strukturalari yanada moslashuvchan, shuning uchun imperativ dasturlashda eski versiya yo‘qoladi, shunchaki yangisi bilan almashtiriladi, funksional ravishda u avtomatik ravishda mavjud bo‘lib qoladi.

Boshqacha qilib aytganda, imperativ dasturlashda ma’lumotlar strukturalari **vaqtinchalik (ing. ephemeral)**, funksional dasturlarda ular odatda **doimiydir**.

Abstrakt ma’lumotlar turi

- **Abstrakt ma’lumotlar turi (ADT - Abstract Data Type)** - bu ma’lumotlar turlari uchun matematik model, bu yerda ma’lumotlar turi xatti-harakatlari (semantikasi) bilan foydalanuvchi nuqtai nazaridan aniqlanadi, ya’ni mumkin bo‘lgan qiymatlar, ushbu ma’lumotlar bo‘yicha mumkin bo‘lgan amallar turi va ushbu amallarning harakati.
- Rasmiy ravishda, ADTni komponentalar ro‘yxati bilan belgilanadigan obyektlar to‘plami (bu obyektlarga taalluqli amallar va ularning xususiyatlari) deb ta’riflash mumkin.

Stek



- **Stek** - Stack inglizchadan uyum, g‘aram, dasta, bog‘lam degan ma’noni anglatadi.
- **Stek** - bu LIFO (last in - first out; oxirgi kelgan - birinchi ketadi) prinsipi bo‘yicha ishlaydigan ma’lumotlar strukturasi.

C ++ tilida stekni realizatsiya qilish

Dastur boshida stek shablonidan foydalanish uchun **<stack>** kutubxonasini yoqishimiz kerak.

#include <stack> //stek kutubxonasini ulash

Stek yaratish uchun biz quyidagi sxema bilan ishlashimiz kerak:

stack <ma'lumot_turi> <nom>;

Yangi satrda **stack** kalit so'zini yozishimiz kerak.

<ma'lumotlar turi> - bu yerda stekda saqlanadigan ma'lumotlar turini yozishimiz kerak.

<nom> - bu stek nomi.

Navbat

- Navbat - bu FIFO (First In - First Out - "birinchi kelgan - birinchi ketadi") prinsipi bo‘yicha qurilgan ma’lumotlar strukturasi.
- Navbatda, agar siz avval kiritilgan elementni q o‘shsangiz, u birinchi bo‘lib chiqadi.

Queue Data Structure

C++ tilida navbatni realizatsiya qilish

Agar siz C++da navbat shablonidan foydalanmoqchi bo‘lsangiz, unda avval **<queue>** kutubxonasini kiritishingiz kerak.

#include <queue> // Queue kutubxonasini ulash

Bundan tashqari, navbatni e’lon qilish uchun quyidagi strukturani ishlatishingiz kerak.

queue <ma’lumot turi> <nom>;

Misol uchun:

queue <int> navbat;

Navbatning metodlari.

Navbat bilan ishlash uchun **push(), pop(), front(), back(), empty()** funksiyalarni bilish kerak**.**

1. Navbatga yangi element qo‘shish uchun **push()** funksiyasidan foydalanish kerak. Qavslar tarkibida biz qo‘shmoqchi bo‘lgan qiymat bo‘lishi kerak.

2. Agar biz birinchi elementni olib tashlashimiz kerak b o‘lsa, **pop()** funksiyasi bilan ishlashimiz kerak. Qavslar ichida endi ko‘rsatilishi kerak bo‘lgan narsa yo‘q, lekin qoidalarga ko‘ra, ular albatta mavjud bo‘lishi kerak. Ushbu funksiyalarga argument kerak emas: **empty(), back() va front().**

3. Agar navbatning birinchi elementiga murojaat qilishingiz kerak bo‘lsa, unda **front()** funksiyasi kerak.

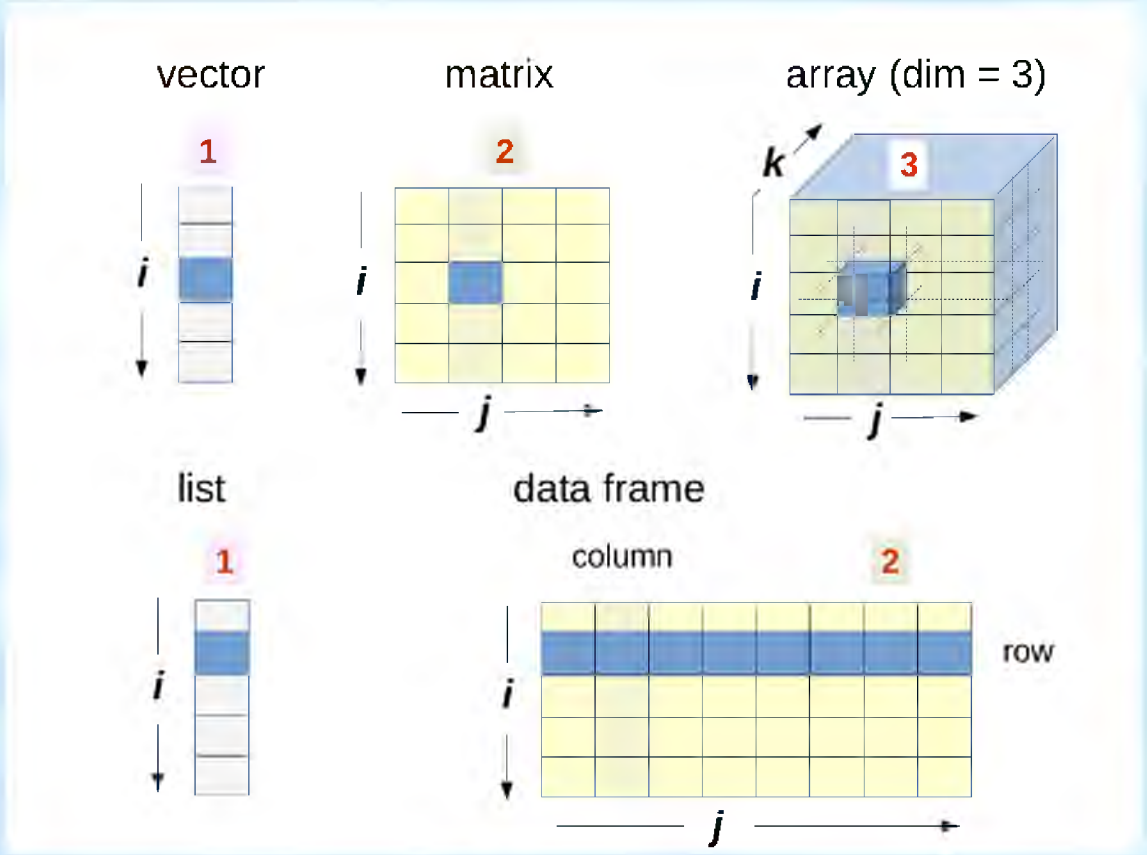
4. **back()** funksiyasi navbatdagi oxirgi elementga kirishga yordam beradi.

5. Navbatning bo‘shligini bilish uchun **empty()** funksiyasidan foydalanish mumkin.

- Agar sizning navbatingiz bo‘sh bo‘lsa, u **true** qiymatini qaytaradi.
- Agar unda biror narsa bo‘lsa, u **false** qaytadi.

Vektor

- **Vektor** - bu dinamik massiv modeli bo‘lgan ma‘lumotlar strukturasi.



C++ tilida vektorlar yaratish

Birinchi navbatda vektorlar yaratish uchun **<vector>** kutubxonasini bog‘lash kerak.

#include <vector> //kutubxonani ulash

Xuddi stek va navbat konstruksiyasi kabi u ham quyidagicha e‘lon qilinadi:

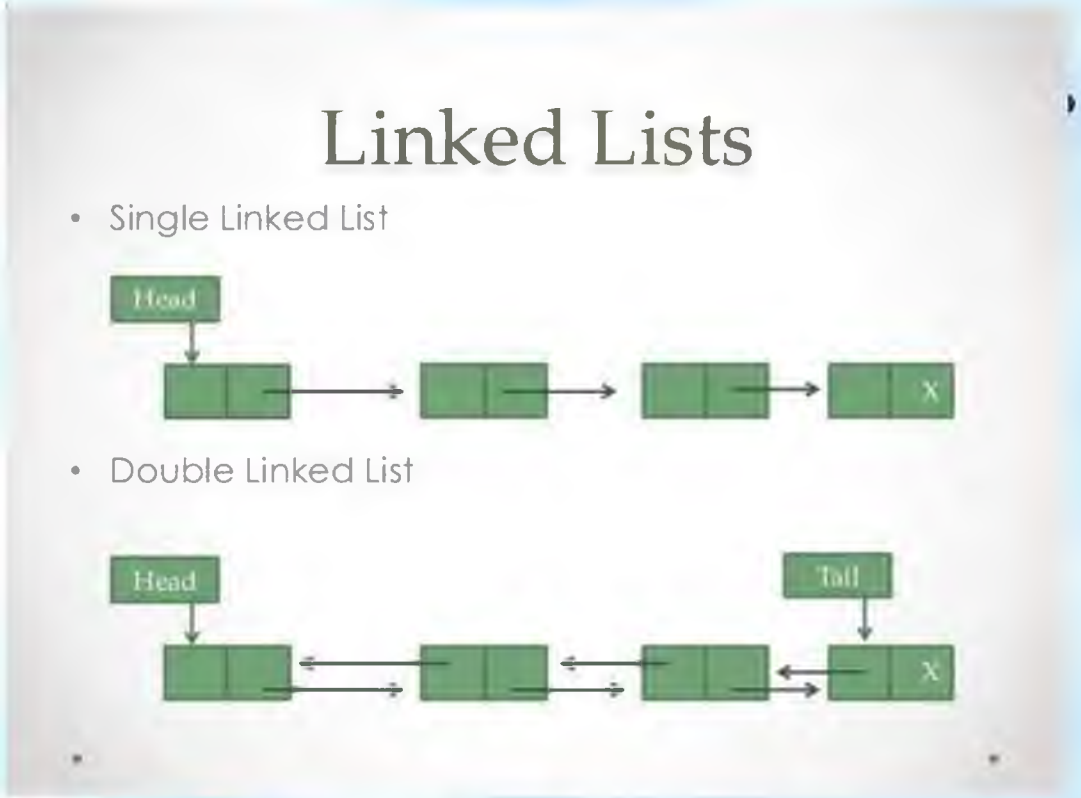
vector <ma’lumot turi> <VektorNomi>

Bundan tashqari vektorga boshlang‘ich qiymatlar berishingiz mumkin.

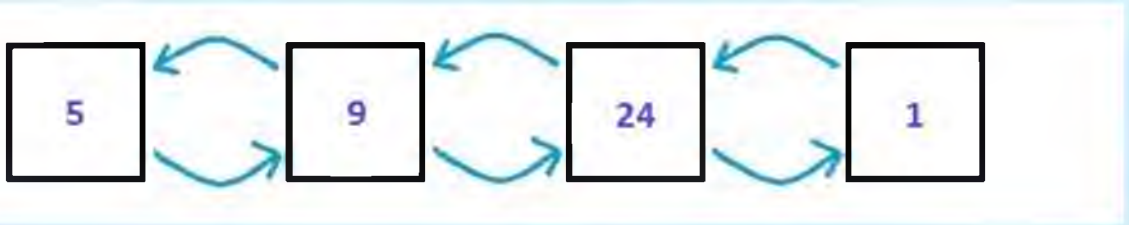
Masalan:

vector <int> V = {7, 4, 3};

Ro‘yxat



- **Ro‘yxat** - bu ikki tomonlama bog‘langan ro‘yxatlarga asoslangan ma‘lumotlar strukturasi. Bu shuni anglatadiki, har qanday element faqat oldingi va keyingi elementlar haqida biladi.



C++ tilida ro‘yxat hosil qilish

Dastlab **list** kutubxonasini ulash lozim.

#include <list>

Oldingi konstruktorlar kabi ro‘yxatni e’lon qilamiz:

list <ma’lumot_turi> <Ro‘yxat_nomi>

Masalan:

list <int> L = {4, 6, 3, 2}

Ro‘yxat bilan ishlash metodlari:

Funksiya nomi	Tavsif
pop_front()	Boshlang‘ich elementini o‘chirish
pop_back()	Oxirgi elementini o‘chirish
push_front()	Boshidan element qo‘shish
push_back()	Oxiridan element qo‘shish
front()	Birmchi elementiga murojaat
back()	Oxirgi elementiga murojaat
insert()	Ko‘rsatilgan joyga element qo‘shish
unique()	Barcha dublikatlarni o‘chirish
merge()	boshqa ro‘yxatni qo‘shish

Mavzu yuzasidan savollar:

1. Ma‘lumot strukturasi tushunchasi nimani anglatadi?
2. Abstrakt ma‘lumotlar strukturasi haqida gapiring.
3. Stek ma‘lumotlar strukturasiga doir misollar keltiring.
4. Navbat va ro‘yxat ma’lumotlar strukturasi bir-biridan qanday farq qiladi?
5. Vektor ma‘ lumotlar strukturasi uchun aniqlangan metodlarni ko‘rsating.

Do you have any questions?