11

O‘N BIRINCHI MA’RUZA

Dasturlash 1 · C++ · 7-mavzu

Ko‘rsatkichlar va dinamik xotira

& va * · ko‘rsatkich arifmetikasi · havolalar va funksiyalar · new / delete · dinamik massivlar · aqlli ko‘rsatkichlar

int *p | &x, *p | new / delete | unique_ptr

ISINISH

Do‘stingizga kitobni berasizmi yoki kitob turgan javon manzilini?

Katta narsani nusxalash qimmat — manzilni aytish esa arzon.

1

Manzilni bilgan odam kitobni o‘zgartira oladimi?

2

Manzil noto‘g‘ri yoki eskirgan bo‘lsa nima bo‘ladi?

3

Kutubxonadan olgan kitobni qaytarmasangiz-chi?

O‘QUV MAQSADLARI

Bu ma’ruza oxirida siz quyidagilarni bajara olasiz...

1. **Tushuntirish** — xotira manzili va ko‘rsatkich tushunchasini
2. **Qo‘llash** — & (manzil olish) va * (murojaat) operatorlarini
3. **Bog‘lash** — massiv va ko‘rsatkich: a[i] ≡ *(a + i)
4. **Farqlash** — qiymat, ko‘rsatkich va havola orqali uzatishni
5. **Ajratish** — dinamik xotirani new/new[] bilan va bo‘shatishni delete/delete[] bilan
6. **Qurish** — kengayuvchi va ikki o‘lchovli dinamik massivlarni
7. **Aniqlash** — xotira sizishi, osilib qolgan ko‘rsatkich, double free xatolarini
8. **Foydalanish** — unique_ptr, shared_ptr va vector dan

01

Ko‘rsatkich tushunchasi

Manzil · int *p = &x · *p orqali murojaat · nullptr

&x

*p

nullptr

8 bayt

Qiymat emas — manzil saqlaydigan o‘zgaruvchi

Ko‘rsatkich (pointer) — qiymat sifatida boshqa o‘zgaruvchining xotiradagi manzilini saqlaydigan o‘zgaruvchi.

&x — manzil olish

x o‘zgaruvchining xotiradagi manzilini qaytaradi: `int *p = &x;`

*p — murojaat

p ko‘rsatgan joydagi qiymat: `cout << *p;`, `*p = 20;`

nullptr

Hech narsani ko‘rsatmaydigan ko‘rsatkich: `int *q = nullptr;`

* ikki ma’noda: e’londa (`int *p`) — “bu ko‘rsatkich”, ifodada (`*p = 20`) — “ko‘rsatilgan joydagi qiymat”.

AMALDA

Ko‘rsatkich orqali o‘zgartirish

```cpp
int x = 10;
int *p = &x;
cout << "x = " << x << ", *p = " << *p << endl;
*p = 25;
cout << "x = " << x << endl;
cout << (p == &x) << endl;
cout << "sizeof(p) = " << sizeof(p);
```

▶ NATIJA

```
x = 10, *p = 10
x = 25
1
sizeof(p) = 8
```

`*p = 25`

x ni to‘g‘ridan-to‘g‘ri emas, p orqali o‘zgartirdik

`p == &x`

p ichida aynan x ning manzili (1 — rost)

`sizeof(p)`

64 bitli tizimda istalgan ko‘rsatkich 8 bayt

nullptr va turli turdagi ko‘rsatkichlar

```
int a = 5, b = 8;
double d = 3.14;
int *p = nullptr;
if (p == nullptr) cout << "bo'sh" << endl;
p = &a;
cout << *p << endl;
p = &b;
*p += 10;
cout << b << endl;
double *pd = &d;
cout << *pd;
```

▶ NATIJA

```
bo'sh
5
18
3.14
```

`p = &b`

ko‘rsatkich boshqa o‘zgaruvchiga yo‘naltirildi

`*p += 10`

b = 8 + 10 = 18

`int *q = &d;`

XATO: `int*` faqat `int` ni ko‘rsatadi

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

1-bo‘lim yakuni: Ko‘rsatkich tushunchasi

1

`int x = 10; int *p = &x; *p = 20;` dan keyin x = ?

Savol 1

2

`&` va `*` operatorlarining vazifasi nima?

Savol 2

3

nullptr nima uchun kerak?

Savol 3

AMALIY TOPSHIRIQ

Har kim o‘z qurilmasida · 10 daqiqa · Oson · 10 ball

1-topshiriq: Ko‘rsatkich orqali almashtirish

`void almashtir(int* a, int* b)` funksiyasini yozing: u ko‘rsatkichlar orqali ikki o‘zgaruvchining qiymatlarini almashtirsin. `main()` tayyor: ikki son o‘qiydi, funksiyani chaqiradi va natijani chiqaradi.

Kirish: Ikki butun son `x` va `y`.

Chiqish: Almashtirilgandan keyingi `x y` — bitta qatorda.

MISOL · KIRISH	MISOL · CHIQISH
`3 8`	`8 3`

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. Kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

02

Ko‘rsatkich arifmetikasi

Massiv nomi = manzil · p + i · *(p + i) ≡ a[i] · q - p

a == &a[0]	p + 1	*(p + i)	q - p

Ko‘rsatkich elementlar bo‘yicha siljiydi

Ifoda	Ma’nosi
`p + i`	i-elementning manzili (`&a[i]`)
`*(p + i)`	i-element qiymati — `a[i]` bilan aynan bir xil
`p++, p--`	Keyingi / oldingi elementga o‘tish
`q - p`	Ikki ko‘rsatkich orasidagi elementlar soni
`p < q`	Solishtirish (bitta massiv ichida)

`p + 1` manzilni 1 baytga emas, `sizeof(tur)` baytga oshiradi: int* — 4, double* — 8, char* — 1.

A[I] ≡ *(P + I)

Massivni ko‘rsatkich bilan aylanib chiqish

```
int a[5] = {10, 20, 30, 40, 50};
int *p = a;
cout << *(p + 2) << " " << p[3] << endl;
for (int *q = a; q < a + 5; q++)
    cout << *q << " ";
cout << endl;
int *oxir = &a[4];
cout << oxir - p << " ta element oraliq";
```

▶ NATIJA

```
30 40
10 20 30 40 50
4 ta element oraliq
```

int *p = a

massiv nomi — birinchi element manzili

p[3]

ko‘rsatkich bilan ham [] ishlaydi

q < a + 5

a + 5 — oxirgi elementdan keyingi joy

oxir - p

elementlar farqi: 4

Massivni joyida teskari aylantirish

```
int a[] = {4, 17, 9, 23, 5};
int *chap = a, *ong = a + 4;
while (chap < ong) {
    int t = *chap;
    *chap = *ong;
    *ong = t;
    chap++;
    ong--;
}
for (int x : a) cout << x << " ";
```

▶ NATIJA

```
5 23 9 17 4
```

chap, ong

ikki uchdan markazga qarab harakat

***chap ⇔ *ong**

qiymatlar almashadi, ko‘rsatkichlar yaqinlashadi

chap < ong

uchrashganda to‘xtaydi

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

2-bo‘lim yakuni: Ko‘rsatkich arifmetikasi

1

`int a[5]; int *p = a;` bo‘lsa `*(p + 3)` nima?

Savol 1

2

double* ko‘rsatkichga 1 qo‘shilsa manzil necha baytga oshadi?

Savol 2

3

`*p + 1` va `*(p + 1)` farqi?

Savol 3

AMALIY TOPSHIRIQ

Har kim o‘z qurilmasida · 10 daqiqa · Oson · 10 ball

2-topshiriq: Ko‘rsatkich bilan yurish

`long long yigindi(const int* bosh, const int* oxir)` funksiyasini yozing: u `[bosh, oxir)` oralig‘idagi elementlar yig‘indisini **indekssiz**, faqat ko‘rsatkichni oshirib hisoblasin. `main()` butun massiv va uning ikkinchi yarmi uchun chaqiradi.

Kirish: Birinchi qatorda `n` ($1 \le n \le 1000$), ikkinchi qatorda `n` ta butun son ($|son| \le 10^9$).

Chiqish: Ikki qator: barcha elementlar yig‘indisi va `a + n/2` dan oxirigacha bo‘lgan elementlar yig‘indisi.

MISOL · KIRISH	MISOL · CHIQISH
4 1 2 3 4	10 7

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. Kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

03

Havolalar va funksiyalar

Qiymat, ko‘rsatkich va havola orqali uzatish · const va ko‘rsatkich · int **pp

int &x

swap

const int *

int **

Funksiya asl o‘zgaruvchini o‘zgartira oladimi?

Usul	E’lon	Chaqiruv	Asl o‘zgaradimi?
Qiymat bo‘yicha	`void f(int x)`	`f(a)`	Yo‘q — nusxa
Ko‘rsatkich orqali	`void f(int *x)`	`f(&a)`	Ha, `*x` orqali
Havola orqali	`void f(int &x)`	`f(a)`	Ha, x — a ning taxallusi

Havola (`int &r = a;`) — mavjud o‘zgaruvchining ikkinchi nomi. U nullptr bo‘lolmaydi va boshqa o‘zgaruvchiga qayta bog‘lanmaydi.

UCH XIL SWAP

Qaysi biri haqiqatan almashtiradi?

```
void s1(int x, int y) { int t = x; x = y; y = t; }
void s2(int *x, int *y) { int t = *x; *x = *y; *y = t; }
void s3(int &x, int &y) { int t = x; x = y; y = t; }
int main() {
    int a = 5, b = 9;
    s1(a, b);  cout << a << b << " ";
    s2(&a, &b); cout << a << b << " ";
    s3(a, b);  cout << a << b;
}
```

▶ NATIJA

```
59 95 59
```

s1

nusxalar almashdi — a, b o‘zgarmadi: 59

s2(&a, &b)

manzillar orqali — almashdi: 95

s3(a, b)

havola orqali — yana almashdi: 59

Nimani o‘zgartirish mumkin emas?

const qayerda turibdi?

- `const int *p` — qiymat o‘zgarmas: `*p = 5` ✗, `p = &y` ✓
- `int *const p` — ko‘rsatkich o‘zgarmas: `*p = 5` ✓, `p = &y` ✗
- `const int *const p` — ikkalasi ham o‘zgarmas

Ko‘rsatkichga ko‘rsatkich

- `int **pp = &p;` — ko‘rsatkichning manzili
- `**pp = 100;` — ikki marta murojaat, oxirgi qiymatga
- Ikki o‘lchovli dinamik massivda kerak (5-bo‘lim)

Qoida: `const` o‘zidan chapdagi narsaga (chapda hech narsa bo‘lmasa — o‘ngdagiga) tegishli.

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

3-bo‘lim yakuni: Havolalar va funksiyalar

1

Qaysi swap a va b ni almashtiradi: `swap(int, int)` yoki `swap(int*, int*)`?

Savol 1

2

Havola va ko‘rsatkichning bitta farqini ayting.

Savol 2

3

`const int *p` da nimani o‘zgartirib bo‘lmaydi?

Savol 3

AMALIY TOPSHIRIQ

Har kim o‘z qurilmasida · 10 daqiqa · Oson · 10 ball

3-topshiriq: Havola orqali min va max

`void minmax(const int* a, int n, int& mn, int& mx)` funksiyasini yozing: u massivning eng kichik va eng katta elementini **havola** (`&`) parametrlar orqali qaytarsin.

Kirish: Birinchi qatorda `n` ($1 \le n \le 1000$), ikkinchi qatorda `n` ta butun son.

Chiqish: Bitta qatorda: minimum va maksimum.

MISOL · KIRISH	MISOL · CHIQISH
5 3 -1 8 0 8	-1 8

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. Kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

04

Dinamik xotira: new va delete

Stek va uyum · new, new[] · delete, delete[] · xotira sizishi

Stack

Heap

new[]

delete[]

Xotiraning ikki hududi

	Stek (stack)	Uyum (heap)
Ajratish	Avtomatik, e’lon qilinganda	Qo‘lda, `new` orqali
Bo‘shatish	Avtomatik, blok tugaganda	Qo‘lda, `delete` orqali
Hajmi	Kichik (1–8 MB)	Katta (operativ xotira hajmicha)
O‘lcham	Kompilyatsiya vaqtida ma’lum	Ishlash vaqtida aniqlanadi

Oltin qoida: har bir `new` ga bitta `delete`, har bir `new[]` ga bitta `delete[]`. Bo‘shatilmagan xotira — xotira sizishi (memory leak).

Bitta obyekt va massiv

```
int *p = new int;          // qiymati noaniq
int *q = new int(42);      // 42 bilan
int *r = new int[5]{1, 2, 3, 4, 5};
*p = 7;
cout << *p << " " << *q << endl;
for (int i = 0; i < 5; i++) cout << r[i];
delete p;
delete q;
delete[] r;
p = q = r = nullptr;
```

▶ NATIJA

```
7 42
12345
```

`new int(42)`

bitta int, boshlang‘ich qiymat bilan

`new int[5]{...}`

massiv, initsializatsiya bilan

`delete[] r`

massiv — albatta `[]` bilan

`= nullptr`

osilib qolgan ko‘rsatkichdan saqlanish

O‘lcham faqat ishlash vaqtida ma’lum

```
int n;
cin >> n;
double *baho = new double[n];
double sum = 0;
for (int i = 0; i < n; i++) {
    cin >> baho[i];
    sum += baho[i];
}
cout << "O'rtacha: " << sum / n;
delete[] baho;
baho = nullptr;
```

▶ NATIJA

```
O'rtacha: 80.75
```

new double[n]

n foydalanuvchidan o‘qildi — oddiy massivda bunday bo‘lmaydi

baho[i]

oddiy massiv kabi ishlatiladi

delete[] baho

ish tugagach xotira qaytariladi

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

4-bo‘lim yakuni: Dinamik xotira: new va delete

1

`int *p = new int[n];` qanday bo‘shatiladi?

Savol 1

2

Xotira sizishi nima?

Savol 2

3

Stekning uyumdan ikki farqini ayting.

Savol 3

05

Dinamik massivlar

Sig‘imni ikki barobar oshirish · int **m — dinamik matritsa

capacity | ×2 | int **m | Bo‘shatish tartibi

KENGAYTIRISH

Joy yetmasa — kattaroq blokka ko‘chish

1

Yangi blok

```
int *b = new int[cap * 2];
```

2

Nusxalash

```
for (i < size) b[i] = a[i];
```

3

Eskisini bo‘shatish

```
delete[] a;
```

4

Ko‘rsatkichni yangilash

```
a = b; cap *= 2;
```

Sig‘im har safar ikki barobar oshirilsa, nusxalashlar kam bo‘ladi — `vector` ham aynan shunday ishlaydi.

KOD

0 kiritilguncha kengayuvchi massiv

```
int cap = 2, size = 0, x;
int *a = new int[cap];
while (cin >> x && x != 0) {
    if (size == cap) {
        int *b = new int[cap * 2];
        for (int i = 0; i < size; i++) b[i] = a[i];
        delete[] a;
        a = b;
        cap *= 2;
        cout << "[cap=" << cap << "] ";
    }
    a[size++] = x;
}
cout << size << " ta element";
delete[] a;
```

▶ NATIJA

```
[cap=4] [cap=8] 5 ta element
```

`size == cap`

joy tugadi — kengaytirish vaqti

`a[size++] = x`

yozish va keyin size ni oshirish

2 → 4 → 8

5 ta element uchun ikki marta kengaydi

Ko‘rsatkichlar massivi: int **m

```
int n = 2, k = 3;
int **m = new int*[n];
for (int i = 0; i < n; i++)
    m[i] = new int[k];
for (int i = 0; i < n; i++)
    for (int j = 0; j < k; j++)
        m[i][j] = i * 10 + j;
cout << m[1][2] << endl;
for (int i = 0; i < n; i++)
    delete[] m[i];
delete[] m;
```

▶ NATIJA

12

new int*[n]

avval qatorlarga ko‘rsatkichlar massivi

m[i] = new int[k]

har bir qator — alohida blok

Bo‘shatish

teskari tartibda: avval qatorlar, keyin m

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

5-bo‘lim yakuni: Dinamik massivlar

1

Dinamik massivni kengaytirishning 4 qadamini ayting.

Savol 1

2

`int **m` matritsa qanday tartibda bo‘shatiladi?

Savol 2

3

Nima uchun sig‘im 1 ga emas, 2 barobar oshiriladi?

Savol 3

AMALIY TOPSHIRIQ

Har kim o‘z qurilmasida · 10 daqiqa · O‘rta · 20 ball

4-topshiriq: Dinamik massiv: prefiks yig‘indilar

`n` ta son berilgan, `n` oldindan noma’lum (10^5 gacha). `new` bilan dinamik massiv yarating, sonlarni o‘qing va **ikkinchi** dinamik massivda prefiks yig‘indilarni hisoblang: `p[i] = a[0] + … + a[i]`. Oxirida `delete[]` bilan xotirani bo‘shating.

Kirish: Birinchi qatorda `n` ($1 \le n \le 100000$), ikkinchi qatorda `n` ta butun son ($|son| \le 10^9$).

Chiqish: Bitta qatorda `n` ta prefiks yig‘indi.

MISOL · KIRISH

```
5
1 2 3 4 5
```

MISOL · CHIQISH

```
1 3 6 10 15
```

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. Kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

06

Xotira xatolari va aqlli ko‘rsatkichlar

Leak · dangling · double free · unique_ptr · shared_ptr · vector

Leak | Dangling | unique_ptr | vector

Kompilyator ko‘pincha sezmaydigan xatolar

1 Xotira sizishi

Ajratilgan blok manzili yo‘qotiladi va hech qachon bo‘shatilmaydi.

2 Osilib qolgan ko‘rsatkich

Bo‘shatilgan yoki mavjud bo‘lmagan xotirani ko‘rsatadi.

3 Ikki marta delete

Bitta blok ikki marta bo‘shatiladi — dastur buziladi.

4 new[] / delete

Massiv qavssiz `delete` bilan bo‘shatiladi — noto‘g‘ri.

5 nullptr ga murojaat

`*p`, bunda p == nullptr — “Segmentation fault”.

6 Yechim

Aqlli ko‘rsatkichlar va `vector` — xotira o‘zi bo‘shaydi.

<MEMORY>

unique_ptr va shared_ptr

```cpp
#include <iostream>
#include <memory>
using namespace std;

int main() {
    unique_ptr<int> u = make_unique<int>(10);
    unique_ptr<int> u2 = move(u);
    cout << (u ? "bor" : "bo'sh") << " " << *u2 << endl;
    shared_ptr<int> s1 = make_shared<int>(99);
    {
        shared_ptr<int> s2 = s1;
        cout << s1.use_count() << " ";
    }
    cout << s1.use_count();
    return 0;
}
```

▶ NATIJA

```
bo'sh 10
2 1
```

unique_ptr

yagona ega; nusxalab bo‘lmaydi, faqat `move()`

shared_ptr

bir nechta ega; `use_count()` hisoblagichi

{ ... }

s2 blokdan chiqqach yo‘qoladi — hisoblagich 1

delete yo‘q!

oxirgi ega yo‘qolganda xotira avtomatik bo‘shaydi

Eng xavfsiz dinamik massiv

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    vector<int> v;
    for (int x : {3, 1, 4, 1, 5})
        v.push_back(x);
    v.pop_back();
    int s = 0;
    for (int x : v) s += x;
    cout << v.size() << " " << s;
    return 0;
}
```

▶ NATIJA

```
4 9
```

push_back

oxiriga qo‘shadi, kerak bo‘lsa o‘zi kengayadi

pop_back

oxirgi elementni olib tashlaydi

delete[] kerak emas

vector o‘zi bo‘shaydi

v.at(i)

chegarani tekshiradi

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

6-bo‘lim yakuni: Xotira xatolari va aqlli ko‘rsatkichlar

1

Qaysi aqlli ko‘rsatkichni nusxalab bo‘lmaydi?

Savol 1

2

Osilib qolgan ko‘rsatkich qanday paydo bo‘ladi?

Savol 2

3

shared_ptr xotirani qachon bo‘shatadi?

Savol 3

XULOSA

Asosiy xulosalar

1 Ko‘rsatkich

Manzil saqlaydi: `int *p = &x;`, qiymat — `*p`.

2 Massiv = manzil

`a[i]` ≡ `*(a + i)`; `p + 1` — keyingi element.

3 Uzatish

O‘zgartirish kerak bo‘lsa — ko‘rsatkich yoki havola (`int &x`).

4 new / delete

`new[]` ↔ `delete[]`; bo‘shatilmasa — xotira sizishi.

5 Dinamik massiv

Joy yetmasa: yangi blok, nusxa, delete[], ko‘rsatkichni yangilash.

6 Aqlli ko‘rsatkichlar

`unique_ptr`, `shared_ptr`, `vector` — xotira o‘zi bo‘shaydi.

TAKRORLASH

Yakunlashdan oldin sakkizta savol

1. `int *p = &x;` da p nimani saqlaydi?
2. `*p = 5;` nima qiladi?
3. `a[2]` ni ko‘rsatkich bilan qanday yozasiz?
4. Havola orqali uzatish belgisi?
5. `new int[10]` ni bo‘shatish?
6. Dinamik matritsa bo‘shatish tartibi?
7. unique_ptr egaligini qanday uzatish mumkin?
8. `vector` ga element qo‘shish metodi?

LABORATORIYA ISHI

7-laboratoriya ishi: 6 ta dasturlash topshirig‘i

5 **O‘suvchi massiv**
O‘rta · 20 ball

6 **Ikki ko‘rsatkich: teskari so‘z**
O‘rta · 20 ball

7 **Matritsani burish**
O‘rta · 20 ball

8 **Elementni o‘chirish**
O‘rta · 20 ball

9 **Saralangan massivlarni birlashtirish**
O‘rta · 20 ball

10 **Ko‘phadlar ko‘paytmasi**
Qiyin · 30 ball

Telefon kamerasi bilan skanerlang

Kod saytda yoziladi va server tomonidan avtomatik tekshiriladi.

O‘qituvchi kodingizni ko‘rib, baho qo‘yadi.

CHIQISH CHIPTASI

Darsni tark etishdan oldin

Javoblaringizni qog‘ozga yozing va o‘qituvchiga topshiring.

1

`int a = 3; int *p = &a; *p *= 4;` dan keyin a nechaga teng?

2

Qachon new[] o‘rniga vector ishlatgan ma’qul?

3

Bugungi mavzuda eng tushunarsiz qolgan joy qaysi?