17

O‘N YETTINCHI MA’RUZA

Dasturlash 1 · C++ · 12-mavzu

Polimorfizm

Statik va dinamik polimorfizm · operatorlarni qayta yuklash · vtable va vptr · abstrakt sinflar · dynamic_cast · Ochiq/yopiq tamoyili

virtual | vtable | dynamic_cast | OCP

ISINISH

“Ovoz ber!” buyrug‘ini itga, mushukka va sigirga bersangiz nima bo‘ladi?

Buyruq bitta — natija har xil. Kompyuterda buni qanday tashkil etish mumkin?

1

\+ belgisi sonlar va satrlar uchun bir xil ish qiladimi?

2

Kim qaysi “ovoz” ni chiqarishini qachon bilamiz: dastur yozilayotganda yoki ishlayotganda?

3

Hayvonot bog‘iga yangi hayvon qo‘shilsa, eski kodni o‘zgartirish kerakmi?

O‘QUV MAQSADLARI

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

1. **Farqlash** — statik (overloading, template) va dinamik (virtual) polimorfizmni
2. **Qayta yuklash** — arifmetik, taqqoslash, oqim va ++ operatorlarini
3. **Yaratish** — funktor — operator() ga ega obyektni
4. **Tushuntirish** — virtual chaqiruv mexanizmini: vptr va vtable
5. **Oldini olish** — virtual funksiyadagi standart argument tuzog‘ining
6. **Qo‘llash** — abstrakt sinf va shablon metod naqshini
7. **Tekshirish** — turni xavfsiz dynamic_cast va typeid bilan
8. **Loyihalash** — Ochiq/yopiq tamoyiliga mos kengaytiriluvchan kodni

01

Polimorfizm turlari

Statik va dinamik · erta va kech bog‘lanish · tezlik va moslashuvchanlik

Overloading	Template	virtual	Bog‘lanish

IKKI TUR

Funksiya qachon tanlanadi?

	Statik (kompilyatsiya vaqti)	Dinamik (ishlash vaqti)
Vositalar	Funksiya/operator qayta yuklash, shablonlar	virtual + asosiy ko‘rsatkich/havola
Tanlov	Argument turlariga qarab	Obyektning haqiqiy turiga qarab
Bog‘lanish	Erta (statik)	Kech (dinamik)
Tezlik	Qo‘shimcha xarajat yo‘q	Kichik: vtable orqali bilvosita chaqiruv
Moslashuvchanlik	Turlar oldindan ma’lum	Yangi turlar — mavjud kodni o‘zgartirmasdan

Polimorfizm — bitta nom yoki interfeys orqali turli xatti-harakatni ifodalash.

YONMA-YON

Statik va dinamik bog‘lanish

```
struct Asosiy {
    virtual ~Asosiy() = default;
    void oddiy() const { cout << "A.oddiy "; }
    virtual void virt() const { cout << "A.virt "; }
};
struct Hosila : Asosiy {
    void oddiy() const { cout << "H.oddiy "; }
    void virt() const override { cout << "H.virt "; }
};
int main() {
    Hosila h;
    Asosiy &r = h;
    Asosiy a = h;
    r.oddiy(); r.virt();
    a.virt();
}
```

▶ NATIJA

```
A.oddiy H.virt A.virt
```

`r.oddiy()`

statik — havola turi Asosiy&

`r.virt()`

dinamik — obyekt turi Hosila

`a.virt()`

kesilish — a haqiqatan Asosiy

▶ INTERAKTIV

Tanlov qachon amalga oshiriladi?

Har bir holatni kompilyatsiya yoki ishlash vaqtiga joylashtiring. (Interaktiv slaydlarda bosib bajariladi.)

`chop(5) va chop("a") — overloading`

`kattasi<int>(3, 7) — shablon`

`a + b — Kasr operatori`

`Hayvon *p; p->ovoz(); — virtual`

`Shakl &s; s.yuza(); — virtual`

`p->oddiy(); — virtual emas`

`std::function<int(int)> f; f(3);`

`dynamic_cast<It*>(p)`

Kompilyatsiya vaqtida (statik)

Ishlash vaqtida (dinamik)

TUSHUNISHNI TEKSHIRISH

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

1-bo‘lim yakuni: Polimorfizm turlari

1

Qaysi biri dinamik polimorfizm: overloading, template yoki virtual?

Savol 1

2

Virtual bo‘lmagan metod asosiy havola orqali chaqirilsa qaysi versiya ishlaydi?

Savol 2

3

Dinamik polimorfizmning afzalligi va narxi?

Savol 3

AMALIY TOPSHIRIQ

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

1-topshiriq: Musiqa asboblari

`Asbob` asosiy sinfida virtual `string chal() const` metodi bor. `Gitara` (`dzing`), `Baraban` (`bum`), `Doira` (`tak-tum`) sinflari uni qayta aniqlaydi (`override`). `main()` tayyor: asboblarni `Asbob*` orqali chaladi.

Kirish: Birinchi qatorda `n` ($1 \le n \le 100$), keyin `n` ta asbob nomi: `gitara`, `baraban` yoki `doira`.

Chiqish: Har bir asbob uchun bitta qatorda uning ovozi.

MISOL · KIRISH

```
3
gitara
doira
baraban
```

MISOL · CHIQISH

```
dzing
tak-tum
bum
```

Telefon kamerasi bilan skanerlang

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

02

Operatorlarni qayta yuklash

a + b = operator+(a, b) · a’zo yoki do‘st · ++ prefiks/postfiks · funktorlar

operator+ | friend | operator++(int) | operator()

QAYSI USUL?

Operator — maxsus nomli funksiya

Usul	Ko‘rinishi	Qachon
A’zo funksiya	`Kasr& operator+=(const Kasr&)`	Chap operand — sinf obyekti va o‘zgaradi: `+=`, `++`, `[]`, `()`, `=`
Do‘st / erkin	`friend Kasr operator+(const Kasr&, const Kasr&)`	Operandlar teng huquqli: `+ - * / == <` — `2 + a` ham ishlaydi
Oqim operatori	`friend ostream& operator<<(ostream&, const Kasr&)`	Chap operand — cout, shuning uchun a’zo bo‘lolmaydi

Qoidalar: yangi operator yo‘q (`**`); ustuvorlik va operandlar soni o‘zgarmaydi; `::`, `.`, `?:`, `sizeof` qayta yuklanmaydi; `+` ayirish qilmasin!

Kasr: + , == va <<

```
class Kasr {
    int s, m;
public:
    Kasr(int s = 0, int m = 1) : s(s), m(m) {
        int g = gcd(s, m); this->s /= g; this->m /= g;
    }
    friend Kasr operator+(const Kasr &a, const Kasr &b) {
        return Kasr(a.s * b.m + b.s * a.m, a.m * b.m);
    }
    friend bool operator==(const Kasr &a, const Kasr &b) {
            return a.s == b.s && a.m == b.m; }
    friend ostream& operator<<(ostream &o, const Kasr &k) {
            return o << k.s << "/" << k.m; }
};
int main() {
    Kasr a(3, 4), b(5, 6);
    cout << a + b << " " << (Kasr(2, 4) == Kasr(1, 2)) << " "
            << 2 + a;
}
```

▶ NATIJA

```
19/12 1 11/4
```

a + b

3/4 + 5/6 = 38/24 = 19/12

Kasr(2, 4) == Kasr(1, 2)

qisqartirilgach teng — 1

2 + a

do‘st operator: 2 → Kasr(2) yashirin o‘zgartiriladi

Prefiks, postfiks va operator()

```
class Sanoq {
    int v = 0;
public:
    Sanoq& operator++() { ++v; return *this; }
    Sanoq operator++(int) { Sanoq e = *this; ++v; return e; }
    int get() const { return v; }
};
struct Kopaytir {
    int k;
    int operator()(int x) const { return x * k; }
};
int main() {
    Sanoq s;
    cout << (s++).get() << (++s).get() << s.get() << " ";
    Kopaytir uch{3};
    cout << uch(7);
}
```

▶ NATIJA

```
022 21
```

operator++()

prefiks — yangi qiymat (havola)

operator++(int)

postfiks — soxta int, eski nusxani qaytaradi

operator()

funktor: obyekt funksiya kabi chaqiriladi, holat saqlaydi

TUSHUNISHNI TEKSHIRISH

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

2-bo‘lim yakuni: Operatorlarni qayta yuklash

1

`a + b` (Kasr) kompilyator uchun nimani anglatadi?

Savol 1

2

Nima uchun `operator<<` a’zo bo‘la olmaydi?

Savol 2

3

Postfiks `++` prefiksdan qanday farqlanadi?

Savol 3

AMALIY TOPSHIRIQ

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

2-topshiriq: Vektor operatorlari

`Vektor` (x, y — butun sonlar) uchun operatorlarni qayta yuklang: `a + b`, `a - b`, `a * k` (songa ko‘paytirish), `a * b` (skalyar ko‘paytma — son qaytaradi), `a == b`. `main()` tayyor.

Kirish: Bitta qatorda `x1 y1 x2 y2 k` ($|son| \le 10^6$).

Chiqish: Besh qator: `a + b`, `a - b`, `a * k` — `(x, y)` ko‘rinishida; skalyar ko‘paytma; `TENG` yoki `TENG EMAS`.

MISOL · KIRISH

```
1 2 3 4 2
```

MISOL · CHIQISH

```
(4, 6)
(-2, -2)
(2, 4)
11
TENG EMAS
```

Telefon kamerasi bilan skanerlang

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

03

vtable va vptr

Virtual chaqiruv ichkaridan · vptr narxi · standart argument tuzog‘i

vtable | vptr | sizeof | Default arg

MEXANIZM

s->yuza() qanday topiladi?

1

vtable

Virtual metodi bor har bir sinf uchun bitta jadval: har bir virtual metod manzili

2

vptr

Har bir obyekt ichidagi yashirin ko‘rsatkich — o‘z sinfi vtable iga

3

1-qadam

Obyektdan vptr o‘qiladi

4

2-qadam

vtable dan kerakli katak olinadi

5

3-qadam

Shu manzildagi funksiya chaqiriladi

vptr konstruktorda o‘rnatiladi — shuning uchun asosiy konstruktor ichida virtual chaqiruv asosiy versiyaga boradi (4-bo‘lim).

NARXI

vptr obyektni kattalashtiradi

```
struct Oddiy { int x; void f() {} };
struct Virtual { int x; virtual void f() {} };
struct Shakl {
    virtual ~Shakl() = default;
    virtual double yuza() const { return 0; }
};
struct Doira : Shakl {
    double r = 1;
    double yuza() const override { return 3.14 * r * r; }
};
int main() {
    cout << sizeof(Oddiy) << " " << sizeof(Virtual) << endl;
    Shakl *s = new Doira;
    cout << s->yuza();
    delete s;
}
```

▶ NATIJA

```
4 16
3.14
```

sizeof(Oddiy) = 4

faqat int

sizeof(Virtual) = 16

int + 8 baytli vptr + tekislash (64 bit)

s->yuza()

vptr → Doira vtable → Doira::yuza

TUZOQ

Standart argument statik bog‘lanadi

```
struct Printer {
    virtual ~Printer() = default;
    virtual void chop(int n = 1) const { cout << "P" << n
            << " "; }
};
struct Rangli : Printer {
    void chop(int n = 5) const override { cout << "R" << n
            << " "; }
};
int main() {
    Rangli r;
    Printer &p = r;
    r.chop();
    p.chop();
}
```

▶ NATIJA

```
R5 R1
```

p.chop()

funksiya dinamik tanlanadi — Rangli

n = 1

lekin standart qiymat havola turidan — Printer

Qoida

qayta aniqlangan virtual funksiyada default argumentni o‘zgartirmang

TUSHUNISHNI TEKSHIRISH

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

3-bo‘lim yakuni: vtable va vptr

1

Virtual chaqiruv qanday amalga oshiriladi?

Savol 1

2

vptr qayerda saqlanadi, vtable-chi?

Savol 2

3

Virtual funksiyaning standart argumenti qaysi turga qarab tanlanadi?

Savol 3

04

Abstrakt sinflar

Sof virtual · shablon metod naqshi · konstruktordagi virtual chaqiruv

= 0

Template method

Konstruktor

vptr

Umumiy algoritm — asosiy sinfda, qadamlar — hosilada

```
class Xodim {
protected:
    string ism;
public:
    Xodim(string i) : ism(i) {}
    virtual ~Xodim() = default;
    virtual double maosh() const = 0;
    void hisobot() const { cout << ism << ": " << maosh()
            << endl; }
};
struct Oylikchi : Xodim {
    double o;
    Oylikchi(string i, double o) : Xodim(i), o(o) {}
    double maosh() const override { return o; }
};
struct Soatbay : Xodim {
    double st, soat;
    Soatbay(string i, double s, double h) : Xodim(i), st(s),
            soat(h) {}
    double maosh() const override { return st * soat; }
};
int main() {
    vector<unique_ptr<Xodim>> v;
    v.push_back(make_unique<Oylikchi>("Aziza", 6000000));
    v.push_back(make_unique<Soatbay>("Bekzod", 40000, 180));
    for (auto &x : v) x->hisobot();
}
```

▶ NATIJA

```
Aziza: 6e+06
Bekzod: 7.2e+06
```

`hisobot()`

umumiy algoritm — virtual qadamni chaqiradi

`maosh() = 0`

har bir xodim turi o‘zi hisoblaydi

`unique_ptr`

delete yozish shart emas

`6e+06`

cout katta double ni ilmiy ko‘rinishda chiqaradi — `fixed` bilan to‘g‘rilanadi

KONSTRUKTORDA VIRTUAL

Qurilayotgan obyekt hali “hosila” emas

```
struct Asosiy {
    Asosiy() { salom(); }
    virtual ~Asosiy() { salom(); }
    virtual void salom() const { cout << "Asosiy "; }
};
struct Hosila : Asosiy {
    Hosila() { salom(); }
    ~Hosila() { salom(); }
    void salom() const override { cout << "Hosila "; }
};
int main() {
    Hosila h;
    cout << "| ";
}
```

▶ NATIJA

```
Asosiy Hosila | Hosila Asosiy
```

Asosiy()

hosila qism hali yo‘q — vptr Asosiy vtable ida

Hosila()

endi obyekt to‘liq — Hosila versiyasi

Destruktorlar

teskari: ~Asosiy da yana Asosiy versiyasi

TUSHUNISHNI TEKSHIRISH

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

4-bo‘lim yakuni: Abstrakt sinflar

1

Asosiy konstruktor ichida virtual metod chaqirilsa qaysi versiya ishlaydi?

Savol 1

2

Shablon metod naqshining mohiyati?

Savol 2

3

Abstrakt sinfning hosilasi sof virtual metodni aniqlamasa-chi?

Savol 3

AMALIY TOPSHIRIQ

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

3-topshiriq: Abstrakt to‘lov

Abstrakt `Tolov` sinfi: `summa` va sof virtual `double komissiya() const = 0`. Meros sinflar: `Karta` — 1%, `Naqd` — 0, `Otkazma` — 2000 + 0.5%. Har bir to‘lov uchun komissiyani, oxirida jami summa va jami komissiyani chiqaring.

Kirish: Birinchi qatorda `n` ($1 \le n \le 100$), keyin `n` qatorda: tur (`karta`, `naqd`, `otkazma`) va summa.

Chiqish: Har bir to‘lov uchun `tur summa komissiya`, oxirida `Jami: summa komissiya` (summa va komissiya — 2 xona).

MISOL · KIRISH

```
3
karta 100000
naqd 50000
otkazma 1000000
```

MISOL · CHIQISH

```
karta 100000.00 1000.00
naqd 50000.00 0.00
otkazma 1000000.00 7000.00
Jami: 1150000.00 8000.00
```

Telefon kamerasi bilan skanerlang

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

05

dynamic_cast va typeid

Upcast va downcast · xavfsiz tekshiruv · RTTI · virtual metod bilan almashtirish

Upcast | Downcast | nullptr | typeid

Hayvon* haqiqatan It mi?

Vosita	Tekshiradimi?	Noto‘g‘ri tur bo‘lsa
`dynamic_cast<It*>(p)`	Ha, ishlash vaqtida	`nullptr`
`dynamic_cast<It&>(h)`	Ha, ishlash vaqtida	`std::bad_cast` istisnosi
`static_cast<It*>(p)`	Yo‘q — dasturchiga ishonadi	Aniqlanmagan xatti-harakat
`typeid(*p) == typeid(It)`	Haqiqiy turni solishtiradi	false

Upcast (It → Hayvon) — har doim xavfsiz va avtomatik. Downcast (Hayvon → It) — tekshirish kerak. RTTI faqat virtual metodli sinflarda ishlaydi.

AMALDA

Faqat itlarga xos metod

```
struct Hayvon {
    virtual ~Hayvon() = default;
};
struct It : Hayvon {
    void qoriqla() const { cout << "qo'riqlaydi "; }
};
struct Mushuk : Hayvon {};
int main() {
    Hayvon *zoo[] = {new It, new Mushuk, new It};
    for (Hayvon *h : zoo) {
        if (It *it = dynamic_cast<It*>(h)) it->qoriqla();
        else cout << "(it emas) ";
        cout << (typeid(*h) == typeid(Mushuk)) << " ";
    }
    for (Hayvon *h : zoo) delete h;
}
```

▶ NATIJA

```
qo'riqlaydi 0 (it emas) 1 qo'riqlaydi 0
```

`dynamic_cast<It*>`

It bo‘lsa — ko‘rsatkich, aks holda nullptr

`if (It *it = ...)`

e’lon va tekshiruv bitta joyda

`typeid(*h)`

haqiqiy tur — `*h` (obyekt), `h` emas!

MASLAHAT

Tur tekshiruvlari ko‘payib ketsa...

Xato: “agar it bo‘lsa..., agar mushuk bo‘lsa...”

- `if (dynamic_cast<It*>(h)) ... else if (dynamic_cast<Mushuk*>(h)) ...`
- Yangi hayvon qo‘shilsa — barcha shunday joylarni tahrirlash kerak
- Loyihalashdagi kamchilik belgisi

To‘g‘ri: bitta virtual metod

- `virtual void vazifa() const = 0;`
- It: qo‘riqlaydi, Mushuk: sichqon tutadi
- Chaqiruv: `h->vazifa();` — tekshiruvsiz

dynamic_cast — favqulodda vosita; asosiy vosita — virtual funksiyalar.

TUSHUNISHNI TEKSHIRISH

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

5-bo‘lim yakuni: dynamic_cast va typeid

1

p Mushuk ni ko‘rsatsa `dynamic_cast<It*>(p)` = ?

Savol 1

2

Havola bilan dynamic_cast muvaffaqiyatsiz bo‘lsa nima bo‘ladi?

Savol 2

3

static_cast bilan downcast nima uchun xavfli?

Savol 3

AMALIY TOPSHIRIQ

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

4-topshiriq: dynamic_cast: faqat doiralar

Shakllar `Shakl*` ro‘yxatida saqlanadi (`Doira` va `Tortburchak`). `dynamic_cast` yordamida faqat **doiralar**ni ajratib oling: ularning soni va radiuslari yig‘indisini, so‘ng to‘rtburchaklar soni va eng katta yuzasini chiqaring.

Kirish: Birinchi qatorda `n` ($1 \le n \le 100$), keyin `n` qatorda: `doira r` yoki `tortburchak a b` (butun sonlar).

Chiqish: Ikki qator: `Doiralar: soni radiuslar_yigindisi` va `Tortburchaklar: soni eng_katta_yuza` (to‘rtburchak bo‘lmasa yuza 0).

MISOL · KIRISH

```
4
doira 3
tortburchak 2 5
doira 4
tortburchak 3 3
```

MISOL · CHIQISH

```
Doiralar: 2 7
Tortburchaklar: 2 10
```

Telefon kamerasi bilan skanerlang

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

06

Ochiq/yopiq tamoyili

Kengaytirish uchun ochiq, o‘zgartirish uchun yopiq · fabrika · std::function strategiya

OCP | Fabrika | Strategiya | std::function

OPEN/CLOSED

Yangi imkoniyat — yangi kod, eski kod o‘zgarmaydi

Dastur qismlari kengaytirish uchun ochiq, o‘zgartirish uchun yopiq bo‘lishi kerak.

Polimorf sikl

`for (auto &s : sahna) s->chiz();` — yangi shakl qo‘shilsa ham o‘zgarmaydi.

Fabrika

Tur nomidan obyekt yaratuvchi yagona joy: `yarat("doira")`.

Strategiya

Xatti-harakatni almashtiriladigan obyekt yoki funksiya sifatida berish.

Yangi Yulduz shaklini qo‘shish: faqat `class Yulduz : public Shakl` va fabrikaga bitta qator.

FABRIKA

Kengaytiriluvchan sahna

```
struct Shakl {
    virtual ~Shakl() = default;
    virtual double yuza() const = 0;
};
struct Doira : Shakl {
    double r;
    Doira(double r) : r(r) {}
    double yuza() const override { return 3.14 * r * r; }
};
struct Kvadrat : Shakl {
    double a;
    Kvadrat(double a) : a(a) {}
    double yuza() const override { return a * a; }
};
unique_ptr<Shakl> yarat(const string &tur, double x) {
    if (tur == "doira") return make_unique<Doira>(x);
    if (tur == "kvadrat") return make_unique<Kvadrat>(x);
    return nullptr;
}
int main() {
    vector<unique_ptr<Shakl>> sahna;
    for (auto t : {"doira", "kvadrat", "olmos"}) {
        auto s = yarat(t, 2);
        if (s) sahna.push_back(move(s));
        else cout << t << "? ";
    }
    for (auto &s : sahna) cout << s->yuza() << " ";
}
```

▶ NATIJA

```
olmos? 12.56 4
```

`yarat()`

yagona joy — yangi tur uchun faqat shu yer o‘zgaradi

`"olmos"`

noma’lum tur — nullptr

`sahna sikli`

hech qachon o‘zgarmaydi

Merossiz polimorfizm: strategiya

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

int main() {
    vector<pair<string,
            function<double(double)>>> chegirmalar = {
        {"10%", [](double n) { return n * 0.9; }},
        {"-5000", [](double n) { return n - 5000; }},
        {"yo'q", [](double n) { return n; }},
    };
    for (auto &[nom, f] : chegirmalar)
        cout << nom << ": " << f(50000) << endl;
    return 0;
}
```

▶ NATIJA

```
10%: 45000
-5000: 45000
yo'q: 50000
```

function<double(double)>

istalgan chaqiriladigan narsani saqlaydi

lambda

har bir strategiya — bitta qator

auto &[nom, f]

C++17 tuzilmaviy bog‘lash

TUSHUNISHNI TEKSHIRISH

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

6-bo‘lim yakuni: Ochiq/yopiq tamoyili

1

Ochiq/yopiq tamoyili nimani talab qiladi?

Savol 1

2

Fabrika funksiyasining foydasi nimada?

Savol 2

3

Qachon virtual ierarxiya o‘rniga std::function yetarli?

Savol 3

XULOSA

Asosiy xulosalar

1 Ikki tur

Statik — overloading, template, operatorlar; dinamik — virtual + ko‘rsatkich/havola.

2 Operatorlar

`a + b = operator+(a, b);` `<<` — do‘st; postfiks ++ — soxta int.

3 vtable / vptr

Obyektda vptr, sinfda vtable; virtual chaqiruv — bilvosita.

4 Tuzoqlar

Default argument statik; konstruktorda virtual — asosiy versiya.

5 RTTI

`dynamic_cast` — xavfsiz downcast (nullptr/bad_cast); `typeid(*p)`.

6 OCP

Yangi tur — yangi sinf; mavjud kod o‘zgarmaydi.

TAKRORLASH

Yakunlashdan oldin sakkizta savol

1. Overloading qaysi turdagi polimorfizm?
2. `operator<<` ni qanday e’lon qilamiz?
3. Postfiks ++ imzosi?
4. Funktor nima?
5. vptr nimani ko‘rsatadi?
6. Asosiy destruktor ichida virtual chaqiruv qaysi versiyaga boradi?
7. `dynamic_cast<It&>` muvaffaqiyatsiz bo‘lsa?
8. OCP qisqacha?

LABORATORIYA ISHI

12-laboratoriya ishi: 6 ta dasturlash topshirig‘i

5 **Virtual destruktor**
O‘rta · 20 ball

6 **Kompleks sonlar**
O‘rta · 20 ball

7 **Ifoda daraxti**
Qiyin · 30 ball

8 **Ochiq/yopiq: chegirmalar**
O‘rta · 20 ball

9 **Turlar hisoboti**
O‘rta · 20 ball

10 **Qahramonlar jangi**
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

Statik va dinamik polimorfizmga bittadan o‘z misolingizni keltiring.

2

Nima uchun `dynamic_cast` ko‘payib ketishi loyihalash kamchiligi belgisi?

3

vtable mexanizmini 2–3 gap bilan tushuntiring.