13

O‘N UCHINCHI MA’RUZA · 2-QISM

Dasturlash 1 · C++ · 8-mavzu (2 qismdan 2-qism)

Funksiyalar

Ko‘rinish sohasi va yashash vaqti · Rekursiya · Funksiyaga ko‘rsatkichlar va lambda

int f(int x)

Overloading

Rekursiya

Lambda

ISINISH

Matryoshka qo‘g‘irchog‘ini ochishni algoritm sifatida yozing.

O‘tgan ma’ruzada funksiyalar, parametrlar va shablonlarni o‘rgandik.

1

Har bir qadamda nima qilinadi?

2

Jarayon qachon to‘xtaydi?

3

Bu jarayonni funksiyaning o‘zini chaqirishi bilan ifodalash mumkinmi?

O‘QUV MAQSADLARI

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

1. **Tushuntirish** — lokal, global va static o‘zgaruvchilarning ko‘rinishi va yashash vaqtini
2. **Yechish** — masalalarni rekursiya yordamida (faktorial, Fibonachchi, Xanoy)
3. **Foydalanish** — funksiyaga ko‘rsatkichlar va lambda ifodalardan
4. **Taqqoslash** — rekursiv va iterativ (siklli) yechimlarni

04

Ko‘rinish sohasi va yashash vaqti

Lokal · global · static · ::x

Lokal

Global

static

::x

SCOPE

O‘zgaruvchi qayerda ko‘rinadi va qancha yashaydi?

Turi	Qayerda e’lon	Ko‘rinish	Yashash vaqti
Lokal	Funksiya yoki blok `{ }` ichida	Faqat shu blok	Blok tugaguncha
Global	Barcha funksiyalardan tashqarida	Butun fayl	Butun dastur
static lokal	Funksiya ichida, `static` bilan	Faqat shu funksiya	Butun dastur; bir marta initsializatsiya

Lokal o‘zgaruvchi bir xil nomli globalni yashiradi; globalga murojaat — `::x`. Global o‘zgaruvchilardan kam foydalaning.

YASHIRISH

Bir nom — uch xil x

```
int x = 100;
void f() { cout << "f: " << x << endl; }
int main() {
    int x = 5;
    cout << x << " " << ::x << endl;
    {
        int x = 1;
        cout << "blok: " << x << endl;
    }
    cout << x << endl;
    ::x = 200;
    f();
}
```

▶ NATIJA

```
5 100
blok: 1
5
f: 200
```

`int x = 5`

lokal x globalni yashiradi

`::x`

global x ga murojaat

`{ int x = 1; }`

ichki blokning x i — blokdan keyin yo‘qoladi

STATIC

Chaqiruvlar orasida qiymatni saqlash

```
void hisob() {
    int oddiy = 0;
    static int statik = 0;
    oddiy++;
    statik++;
    cout << oddiy << "-" << statik << " ";
}
int yangiId() {
    static int id = 1000;
    return ++id;
}
int main() {
    hisob(); hisob(); hisob();
    cout << endl << yangiId() << " " << yangiId();
}
```

▶ NATIJA

```
1-1 1-2 1-3
1001 1002
```

`int oddiy = 0`

har chaqiruvda qaytadan 0

`static int statik`

faqat birinchi chaqiruvda 0, keyin saqlanadi

`yangiId()`

noyob ID generatori

TUSHUNISHNI TEKSHIRISH

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

4-bo‘lim yakuni: Ko‘rinish sohasi va yashash vaqti

1

static lokal o‘zgaruvchining xususiyati nima?

Savol 1

2

Lokal va global x bir xil nomda bo‘lsa, globalga qanday murojaat qilinadi?

Savol 2

3

`for (int i = 0; ...)` dan keyin i ni ishlatsa bo‘ladimi?

Savol 3

05

Rekursiya

Bazaviy holat · rekursiv qadam · chaqiruvlar steki · Fibonachchi · Xanoy minorasi

n!

Baza

Stek

Xanoy

Funksiya o‘zini o‘zi chaqiradi

Rekursiv yechimda masala o‘zining kichikroq nusxasiga keltiriladi: n! = n × (n − 1)!

Bazaviy holat

Rekursiyasiz javob: `if (n <= 1) return 1;` — to‘xtash sharti.

Rekursiv qadam

Kichikroq argument bilan chaqiruv: `return n * faktorial(n - 1);`

Chaqiruvlar steki

Har chaqiruvga alohida kadr; natijalar teskari tartibda qaytadi.

Bazaviy holat bo‘lmasa yoki unga yetib borilmasa — cheksiz chaqiruv, stack overflow.

KLASSIK MISOLLAR

Faktorial, daraja, raqamlar yig‘indisi

```cpp
long long fakt(int n) {
    if (n <= 1) return 1;
    return n * fakt(n - 1);
}
int fib(int n) {
    if (n < 2) return n;
    return fib(n - 1) + fib(n - 2);
}
int rYig(int n) {
    if (n < 10) return n;
    return n % 10 + rYig(n / 10);
}
int main() {
    cout << fakt(10) << " " << fib(10) << " " << rYig(987654);
}
```

▶ NATIJA

```
3628800 55 39
```

fakt(10)

10! = 3628800

fib(10)

0 1 1 2 3 5 8 13 21 34 55

rYig(987654)

9+8+7+6+5+4 = 39

Diqqat!

fib(25) 240 mingdan ortiq chaqiruv — sikl ancha tez

XANOY MINORASI

Rekursiya eng chiroyli masala

```
void hanoy(int n, char dan, char ga, char yordam) {
    if (n == 0) return;
    hanoy(n - 1, dan, yordam, ga);
    cout << n << ": " << dan << "->" << ga << endl;
    hanoy(n - 1, yordam, ga, dan);
}
int main() {
    hanoy(2, 'A', 'C', 'B');
}
```

▶ NATIJA

```
1: A->B
2: A->C
1: B->C
```

1-qadam

yuqoridagi n-1 diskni yordamchi ustunga

2-qadam

eng katta diskni maqsadga

3-qadam

n-1 diskni katta disk ustiga

2^n – 1

3 disk — 7 qadam, 64 disk — milliardlab yil

TUSHUNISHNI TEKSHIRISH

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

5-bo‘lim yakuni: Rekursiya

1

Har bir rekursiv funksiyada albatta nima bo‘lishi shart?

Savol 1

2

fakt(4) nechta chaqiruv hosil qiladi?

Savol 2

3

Nima uchun fib(n) rekursiv yechimi sekin?

Savol 3

AMALIY TOPSHIRIQ

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

4-topshiriq: Rekursiv raqamlar yig‘indisi

Ikki **rekursiv** funksiya yozing: `int raqamYig(long long n)` — raqamlar yig‘indisi; `int ildiz(long long n)` — raqamli ildiz (bitta raqam qolguncha raqamlar yig‘indisini olish: 9875 → 29 → 11 → 2).

Kirish: Bitta butun son `n` ($0 \le n \le 10^{18}$).

Chiqish: Ikki qator: raqamlar yig‘indisi va raqamli ildiz.

MISOL · KIRISH	MISOL · CHIQISH
9875	29 2

Telefon kamerasi bilan skanerlang

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

06

Funksiyaga ko‘rsatkichlar va lambda

double (*f)(double, double) · funksiya argument sifatida · lambda va ushlash

(*f)()

Callback

[](int x)

[&sum]

Funksiyani o‘zgaruvchida saqlash

```
double qosh(double a, double b) { return a + b; }
double kop(double a, double b) { return a * b; }
double hisobla(double a, double b, double (*amal)(double,
        double)) {
    return amal(a, b);
}
int main() {
    double (*f)(double, double) = qosh;
    cout << f(3, 4) << " ";
    f = kop;
    cout << f(3, 4) << " ";
    cout << hisobla(10, 4, qosh);
}
```

▶ NATIJA

```
7 12 14
```

`double (*f)(double, double)`

ikki double olib double qaytaruvchi funksiya manzili

`f = kop`

boshqa funksiyaga yo‘naltirish

`hisobla(..., qosh)`

funksiya argument sifatida — “callback”

[ushlash](parametrlar) { tana }

Ushlash	Ma’nosi
`[]`	Tashqi o‘zgaruvchilar ishlatilmaydi
`[k]`	k qiymat bo‘yicha (nusxasi) ushlanadi
`[&sum]`	sum havola orqali — lambda uni o‘zgartira oladi
`[=]`	Barcha tashqi o‘zgaruvchilar qiymat bo‘yicha
`[&]`	Barcha tashqi o‘zgaruvchilar havola orqali

Lambda — joyning o‘zida yoziladigan nomsiz funksiya: `auto kv = [](int x) { return x * x; };`

ALGORITMLAR BILAN

sort, count_if, for_each + lambda

```
vector<int> v = {5, -3, 12, 8, -7, 4};
sort(v.begin(), v.end(), [](int a, int b) { return a > b; });
for (int x : v) cout << x << " ";
cout << endl;
int k = 4;
cout << count_if(v.begin(), v.end(), [k](int x) { return x > k;
        });
int sum = 0;
for_each(v.begin(), v.end(), [&sum](int x) { sum += x; });
cout << " " << sum;
```

▶ NATIJA

```
12 8 5 4 -3 -7
3 19
```

`[](int a, int b)`

kamayish tartibi sharti

`[k]`

k nusxasi — 4 dan kattalar: 12, 8, 5

`[&sum]`

havola — tashqi sum o‘zgaradi

TUSHUNISHNI TEKSHIRISH

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

6-bo‘lim yakuni: Funksiyaga ko‘rsatkichlar va lambda

1

Lambdadagi `[&]` nimani bildiradi?

Savol 1

2

```
auto f = [](int x) { return x + 1; }; f(4)
```

= ?

Savol 2

3

Funksiyani boshqa funksiyaga argument qilib berish qanday nomlanadi?

Savol 3

XULOSA

Asosiy xulosalar

1 Overloading va template

Bir nom — turli parametrlar; `template <typename T>` — bitta kod ko‘p tur uchun.

2 Scope

Lokal — blokda, global — hamma joyda, `static` — qiymatni saqlaydi.

3 Rekursiya va lambda

Bazaviy holat shart; lambda — `[ushlash](param) { tana }`.

TAKRORLASH

Yakunlashdan oldin oltita savol

1. static lokal necha marta initsializatsiya qilinadi?
2. Rekursiyaning ikki qismi?
3. `[&sum]` qanday ushlash?
4. Global va lokal o‘zgaruvchi nomi bir xil bo‘lsa, funksiya ichida qaysi biri ishlatiladi?
5. `fakt(0)` nima qaytaradi?
6. Bazaviy holatsiz rekursiya nima bilan tugaydi?

LABORATORIYA ISHI

8-laboratoriya ishi: 6 ta dasturlash topshirig‘i

5 **Hanoy minorasi**
O‘rta · 20 ball

6 **Binomial koeffitsiyent (memo)**
O‘rta · 20 ball

7 **Lambda bilan saralash**
O‘rta · 20 ball

8 **Funksiyalar jadvali**
O‘rta · 20 ball

9 **Shablon: eng kattasi**
O‘rta · 20 ball

10 **Barcha o‘rin almashtirishlar**
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

`n!` ni rekursiv hisoblovchi funksiyani yozing va bazaviy holatni ko‘rsating.

2

Qachon rekursiya o‘rniga sikl ishlatgan ma’qul?

3

Funksiya ichidagi `static int k = 0;` nima uchun kerak bo‘lishi mumkin?