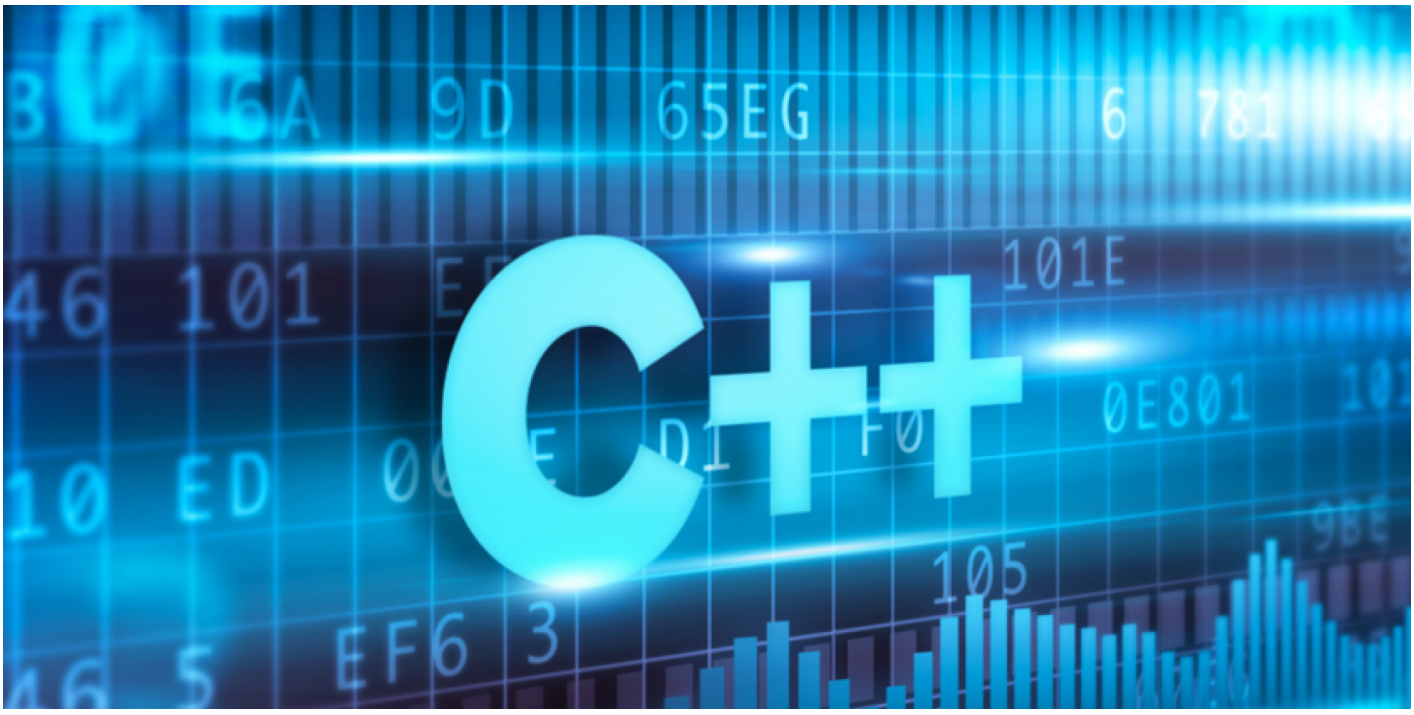
Ko’rsatkichlarga doir masalalarni dasturini tuzishni o’rganish



Ko‘rsatkichga boshlang‘ich qiymat berish

Programma matnida o‘zgaruvchi e’lon qilinganda, kompilyator o‘zgaruvchiga xotiradan joy ajratadi. Boshqacha aytganda, programma kodi xotiraga yuklanganda berilganlar uchun, ular joylashadigan segmentning boshiga nisbatan siljishini, ya’ni nisbiy adresini aniqlaydi va ob’ekt kod hosil qilishda o‘zgaruvchi uchragan joyga uning adresini joylashtiradi.

Umuman olganda, programmadagi o‘zgarmaslar, o‘zgaruvchilar, funksiyalar va sinf ob’ektlar adreslarini xotiraning alohida joyida saqlash va ular ustidan amallar bajarish mumkin. Qiymat-lari adres bo‘lgan o‘zgaruvchilarga *ko‘rsatkich o‘zgaruvchilar* deyiladi.

Ko‘rsatkich uch xil turda bo‘lishi mumkin:

- birorta ob’ektga, xususan o‘zgaruvchiga ko‘rsatkich;
- funksiyaga ko‘rsatkich;
- void ko‘rsatkich.

Ko‘rsatkichning bu xususiyatlari uning qabul qilishi mumkin bo‘lgan qiymatlarida farqlanadi.

Ko‘rsatkich albatta birorta turga bog‘langan bo‘lishi kerak, ya’ni u ko‘rsatgan adresda qandaydir qiymat joylanishi mumkin va bu qiymatning xotirada qancha joy egallashi oldindan ma’lum bo‘lishi shart.

Funksiyaga ko‘rsatkich. Funksiyaga ko‘rsatkich programma joy-lashgan xotiradagi funksiya kodining boshlang‘ich adresini ko‘rsa-tadi, ya’ni funksiya chaqirilganda boshqaruv ayni shu adresga uzatila-di. Ko‘rsatkich orqali funksiyani oddiy yoki vositali chaqirish amalga oshirish mumkin. Bunda funksiya uning nomi bo‘yicha emas, balki funksiyaga ko‘rsatuvchi o‘zgaruvchi orqali chaqiriladi. Funksiyani boshqa funksiyaga argument sifatida uzatish ham funksiya ko‘rsatkichi orqali bajariladi. Funksiyaga ko‘rsatkichning yozilish sintaksisi quyidagicha:

(*) ();

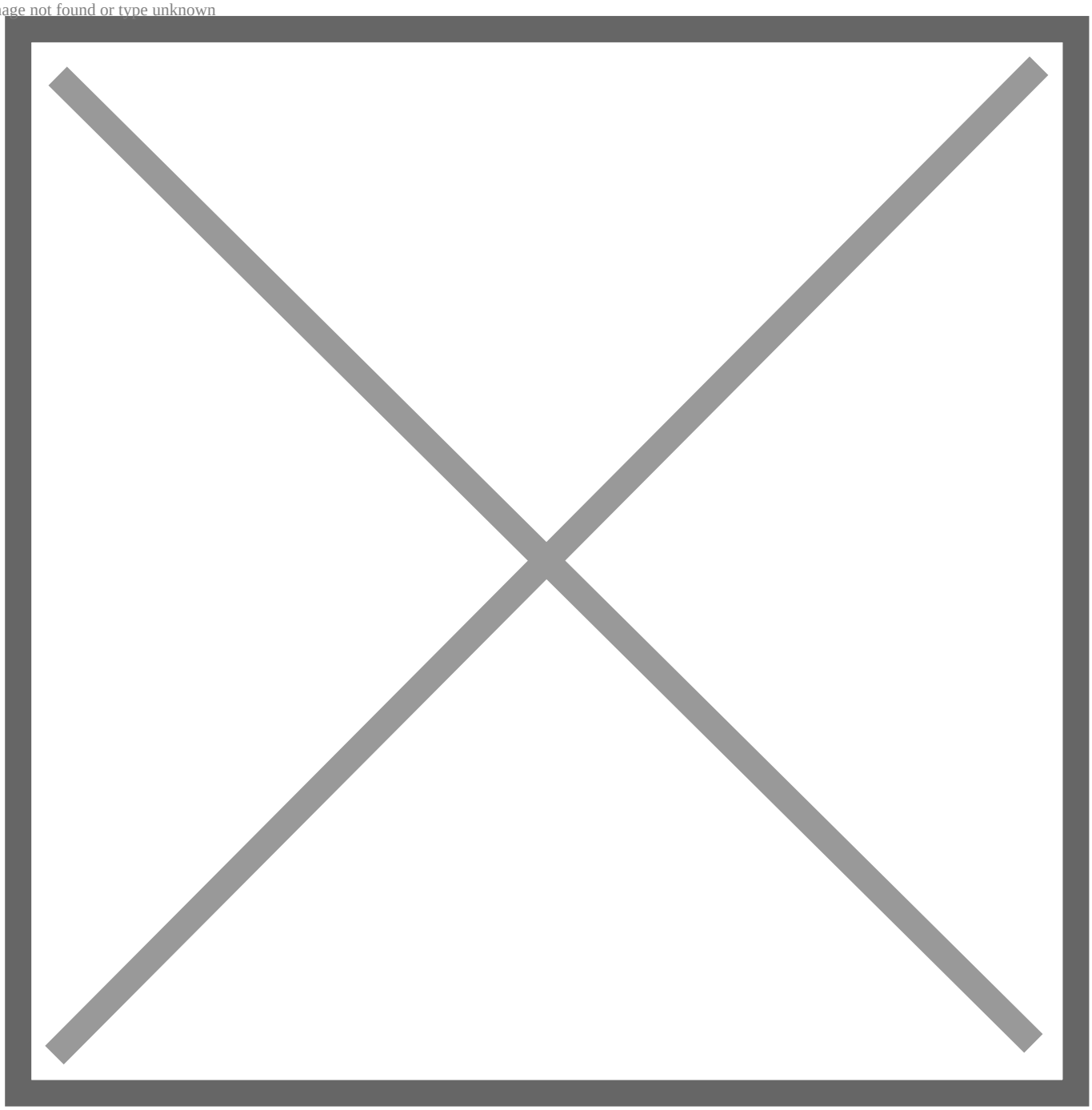
Bunda - funksiya qaytaruvchi qiymat turi; * - ko‘rsatkich o‘zgaruvchining nomi; - funksiya parametr-larining yoki ularning turlarining ro‘yxati.

Masalan:

int (*fun)(float,float);

Bu erda butun son turida qiymat qaytaradigan fun nomidagi funksiyaga ko‘rsatkich e’lon qilingan va u ikkita haqiqiy turdagi parametrlarga ega.

Image not found or type unknown



Programma bosh funksiya, integral hisoblash va ikkita matematik funksiyalar - $f_1(x)$ va $f_3(x)$ uchun aniqlangan funksiyalardan tashkil topadi, funksiyaning adresi «math.h» sarlavha faylidan olinadi. Integral hisoblash funksiyasiga ko‘rsatkich orqali integrali hisoblanadigan funksiya adresi, a va b - integral chegaralari qiymatlari uzatiladi. Oraliqni bo‘lishlar soni - n global o‘zgarmas qilib e’lon qilinadi.

#include

```
#include
const int n=100;
double f1(double x){return 5*sin(3*x)+x;}
double f3(double x){return x*x+1;}
double Integral(double(*f)(double),double a,double b)
{
 double x,s=0;
 double h=(b-a)/n;
 x=a-h/2;
 for(int i=1;i<=n; i++) s+=f(x+=h);
 s*=h;
 return s;
}
int main()
{
 double a,b;
 int menu;
 while(1)
 {
  cout<<"\nIsh regimini tanlang:\n";
  cout<<"1:f1(x)=5*sin(3*x)+x integralini\
  hisoblash\n";
  cout<<"2:f2(x)=cos(x) integralini hisoblash\n";
```

```
 cout<<"3:f3(x)=x^2+1 integralini hisoblash\n";
 cout<<"0:Programmadan chiqish\n";
 do
 {
  cout<<" Ish regimi-> ";
  cin>>menu;
 }
  while (menu<0 || menu>3);
 if(!menu)break;
 cout<<"Integral oralig'ining quyi chegarasi a=";
 cin>>a;
 cout<<"Integral oralig'ining yuqori chegarasi b=";
 cin>>b;
 cout<<"Funksiya integrali S=";
 switch (menu)
 {
  case 1 : cout<<Integral(f1,a,b)<<endl; break;
  case 2 : cout<<Integral(cos,a,b)<<endl; break;
  case 3 : cout<<Integral(f3,a,b)<<endl;
 }
}
 return 0;
}
```

Programmaning ishi cheksiz takrorlash operatori tanasini bajarishdan iborat. Takrorlash tanasida foydalanuvchiga ish reji-mini tanlash bo‘yicha menyu taklif qilinadi:

Ish regimini tanlang:

1: f1(x)=5*sin(3*x)+x integralini hisoblash

2: f2(x)=cos(x) integralini hisoblash

3: f3(x)=x^2+1 integralini hisoblash

0: Programmadan chiqish

Ish regimi->

Ko‘rsatkich ustida amallar

Foydalanuvchi 0 va 3 oralig‘idagi butun sonni kiritishi kerak. Agar kiritilgan son (menu o‘zgaruvchi qiymati) 0 bo‘lsa, break opera-tori yordamida takrorlashdan, keyin programmadan chiqiladi. Agar menu qiymati 1 va 3 oralig‘ida bo‘lsa, integralning quyi va yuqori chegaralarini kiritish so‘raladi, hamda Integral() funksiyasi mos funksiya adresi bilan chaqiriladi va natija chop etiladi. SHunga e’tibor berish kerakki, integral chegaralarining qiymatlarini to‘g‘ri kiritilishiga foydalanuvchi javobgar.

Ob’ektga ko‘rsatkich. Biror ob’ektga ko‘rsatkich (shu jumladan o‘zgaruvchiga). Bunday ko‘rsatkichda ma’lum turdagi (tayanch yoki hosila-viy turdagi) berilganlarning xotiradagi adresi joylashadi. Ob’ektga ko‘rsatkich quyidagicha e’lon qilinadi:

*;

Bu erda - ko‘rsatkich aniqlaydigan adresdagi qiymatning turi, - ob’ekt nomi (identifikator). Agar bir turda bir nechta ko‘rsatkichlar e’lon qilinadigan bo‘lsa, har bir ko‘rsatkich uchun ‘*’ belgisi qo‘yilishi shart:

int *i, j,*k;

float x,*y,*z;

Keltirilgan misolda i va k - butun turdagi ko‘rsatkichlar va j - butun turdagi o‘zgaruvchi, ikkinchi operatorda x - haqiqiy o‘zgaruvchi va y,z - haqiqiy turdagi ko‘rsatkichlar e’lon qilingan.

void ko‘rsatkich. Bu ko‘rsatkich ob’ekt turi oldindan noma’lum bo‘lganda ishlatiladi. void ko‘rsatkichining muhim afzalliklaridan biri - unga har qanday turdagi ko‘rsatkich qiymatini yuklash mumkin-ligidir. void ko‘rsatkich adresidagi qiymatni ishlatishdan oldin, uni aniq bir turga oshkor ravishda keltirish kerak bo‘ladi. void ko‘rsatkichni e’lon qilish kuyidagicha bo‘ladi:

void *;

Ko‘rsatkichning o‘zi o‘zgarmas yoki o‘zgaruvchan bo‘lishi va o‘zgarmas yoki o‘zgaruvchilar adresiga ko‘rsatishi mumkin, masalan:

int i; // butun o‘zgaruvchi

const int ci=1; // butun o‘zgarmas

int * pi; // butun o‘zgaruvchiga ko‘rsatkich

const int *pci; // butun o‘zgarmasga ko‘rsatkich

int *const cp=&i;//butun o‘zgaruvchiga o‘zgarmas

//ko‘rsatkich

const int*const cpc=&ci; // butun o‘zgarmasga o‘zgarmas

// ko‘rsatkich

Misollardan ko‘rinib turibdiki, ‘*’ va ko‘rsatkich nomi ora-sida turgan const modifikatori faqat ko‘rsatkichning o‘ziga tegishli hisoblanadi va uni o‘zgartirish mumkin emasligini bildiradi, ‘*’ belgisidan chapda turgan const esa ko‘rsatilgan adresdagi qiymat o‘zgarmas ekanligini bildiradi.

Ko‘rsatkichga qiymatni berish uchun ‘&’ - adresni olish amali ishlatiladi.

Ko‘rsatkich o‘zgaruvchilarining amal qilish sohasi, yashash davri va ko‘rinish sohasi umumiy qoidalarga bo‘ysunadi.

Ko‘rsatkichga boshlang‘ich qiymat berish

Ko‘rsatkichlar ko‘pincha dinamik xotira (boshqacha nomi «uyum» yoki «heap») bilan bog‘liq holda ishlatiladi. Xotiraning dinamik deyilishiga sabab, bu sohadagi bo‘sh xotira programma ishlash jarayonida, kerakli paytida ajratib olinadi va zarurat qolmaganida qaytariladi (bo‘shatiladi). Keyinchalik, bu xotira bo‘lagi programma tomonidan boshqa maqsadda yana ishlatilishi mumkin. Dinamik xotiraga faqat ko‘rsatkichlar yordamida murojaat qilish mumkin. Bunday o‘zgaruvchilar *dinamik o‘zgaruvchilar* deyiladi va ularni yashash vaqti yaratilgan nuqtadan boshlab programma oxirigacha yoki oshkor ravishda yo‘qotilgan (bog‘langan xotira bo‘shatilgan) joygacha bo‘ladi.

Ko‘rsatkichlarni e’lon qilishda unga boshlang‘ich qiymatlar berish mumkin. Boshlang‘ich qiymat (initsializator) ko‘rsatkich nomi-dan so‘ng yoki qavs ichida yoki ‘=’ belgidan keyin beriladi. Boshlang‘ich qiymatlar quyidagi usullar bilan berilishi mumkin:

1. Ko‘rsatkichga mavjud bo‘lgan ob’ektning adresini berish:
2. a) adresni olish amal orqali:

```
int i=5,k=4; // butun o‘zgaruvchilar

int *p=&i;      // p ko‘rsatkichga i o‘zgaruvchining

// adresi yoziladi

int *p1(&k); // p1 ko‘rsatkichga k o‘zgaruvchining

// adresi yoziladi
```

1. b) boshqa, initsializatsiyalangan ko‘rsatkich qiymatini berish:

```
int * r=p; // p oldin e’lon qilingan va qiymatga ega

                    // bo‘lgan ko‘rsatkich
```

1. v) massiv yoki funksiya nomini berish:

```
int b[10];  // massivni e’lon qilish

int *t=b;  // massivning boshlang‘ich adresini berish

void f(int a){/* ... */} // funksiyani aniqlash

void (*pf)(int);// funksiyaga ko‘rsatkichni e’lon qilish

pf=f; // funksiya adresini ko‘rsatkichga berish
```

1. II. Oshkor ravishda xotiraning absolyut adresini berish:

```
char *vp = (char *)0xB8000000;
```

Bunda 0xB8000000 - o‘n oltilik o‘zgarmas son va (char*) - turga keltirish amali bo‘lib, u vp o‘zgaruvchisini xotiraning absolyut adresidagi baytlarni char sifatida qayta ishlovchi ko‘rsatkich turiga aylantirilishini anglatadi.

III. Bo‘sh qiymat berish:

```
int *suxx=NULL;

int *r=0;
```

Birinchi satrda maxsus NULL o‘zgarmasi ishlatilgan, ikkinchi satrda 0 qiymat ishlatilgan. Ikkala holda ham ko‘rsatkich hech qanday ob’ektga murojaat qilmaydi. Bo‘sh ko‘rsatkich asosan ko‘rsatkichni aniq bir ob’ektga ko‘rsatayotgan yoki yo‘qligini aniqlash uchun ishla-tiladi.

1. Dinamik xotirada new amali bilan joy ajratish va uni adresini ko‘rsatkichga berish:

```
int * n=new int;                    // birinchi operator

int * m=new int(10); // ikkinchi operator

int * q=new int[5];     // uchinchi operator
```

Birinchi operatorda new amali yordamida dinamik xotirada int uchun etarli joy ajratib olinib, uning adresi n ko‘rsatkichga yuklanadi. Ko‘rsatkichning o‘zi uchun joy kompilyasiya vaqtida ajra-tiladi.

6.1-rasm. Dinamik xotiradan joy ajratish

Ikkinchi operatorda joy ajratishdan tashqari m adresiga boshlang‘ich qiymat - 10 sonini joylashtiradi.

Uchinchi operatorda int turidagi 5 element uchun joy ajra-tilgan va uning boshlang‘ich adresi q ko‘rsatkichga berilayapti.

Xotira new amali bilan ajratilgan bo‘lsa, u delete amali bilan bo‘shatilishi kerak. YUqoridagi dinamik o‘zgaruvchilar bilan bog‘lan-gan xotira quyidagicha bo‘shatiladi:

delete n; delete m; delete[]q;

Agarda xotira new[] amali bilan ajratilgan bo‘lsa, uni bo‘shatish uchun delete [] amalini o‘lchovi ko‘rsatilmagan holda qo‘llash kerak.

Xotira bo‘shatilganligiga qaramasdan ko‘rsatkichni o‘zini keyinchalik qayta ishlatish mumkin.

Ko‘rsatkich ustida amallar

Ko‘rsatkich ustida quyidagi amallar bajarilishi mumkin:

1. ob’ektga vositali murojaat qilish amali;
2. qiymat berish amali;
3. ko‘rsatkichga o‘zgarmas qiymatni qo‘shish amali;
4. ayirish amali;
5. inkrement va dekrement amallari;
6. solishtirish amali;
7. turga keltirish amali.

Vositali murojaat qilish amali ko‘rsatkichdagi adres bo‘yicha joylashgan qiymatni olish yoki qiymat berish uchun ishlatiladi:

shar a; // char turidagi o‘zgaruvchi e’loni.

shar *p=new char; // Ko‘rsatkichni e’lon qilib,unga

// dinamik xotiradan ajratilgan

// xotiraning adresini berish

*p=’b’; // p adresiga qiymat joylashtirish

a=*p; // a o‘zgaruvchisiga p adresidagi qiymatni berish

SHuni qayd qilib o‘tish kerakki, xotiraning aniq bir joyidagi adresni bir paytning o‘zida bir nechta va har xil turdagi ko‘rsatkich-larga berish mumkin va ular orqali murojaat qilinganda berilgan-ning har xil turdagi qiymatlarini olish mumkin:

unsigned long int A=0Xcc77ffaa;

unsigned short int * pint=(unsigned short int*)&A;

unsigned char* pchar=(unsigned char*)&A;

cout<<hex<<A<<’ ’<<hex<<*pint<<’ ’<<hex<<(int)*pchar;

Ekranga har xil qiymatlar chop etiladi:

cc77ffaa ffaa aa

O‘zgaruvchilar bitta adresda joylashgan holda yaxlit qiymatning turli bo‘laklarini o‘zlashtiradi. Bunda, bir baytdan katta joy egal-lagan son qiymatining xotirada «teskari» joylashishi inobatga olinishi kerak.

Agar har xil turdagi ko‘rsatkichlarga qiymatlar berilsa, albat-ta turga keltirish amalidan foydalanish kerak:

int n=5;

float x=1.0;

int *pi=&n;

float *px=&x;

void *p;

int *r,*r1;

px=(float *)&n;

p=px;

r=(int *)p;

r1=pi;

Ko‘rsatkich turini void turiga keltirish amalda ma’noga ega emas. Xuddi shunday, turlari bir xil bo‘lgan ko‘rsatkichlar uchun turni keltirish amalini bajarishga hojat yo‘q.

Ko‘rsatkich ustidan bajariladigan arifmetik amallarda avto-matik ravishda turlarning o‘lchami hisobga olinadi.

Arifmetik amallar faqat bir xil turdagi ko‘rsatkichlar ustidan bajariladi va ular asosan, massiv tuzilmalariga ko‘rsatkich-lar ustida bajariladi.

Inkrement amali ko‘rsatkichni massivning keyingi elementiga, dekrement esa aksincha, bitta oldingi elementining adresiga ko‘chiradi. Bunda ko‘rsatkichning qiymati sizeof() qiymatiga o‘zgaradi. Agar ko‘rsatkich k o‘zgarmas qiymatga oshirilsa yoki kamaytirilsa, uning qiymati k*sizeof() kattalikka o‘zgaradi.

Masalan:

short int *p=new short[5];

long * q=new long [5];

p++;   // p qiymati 2 oshadi

q++;   // q qiymati 4 ga oshadi

q+=3; // q qiymati 3*4=12 oshadi

Ko‘rsatkichlarning ayirmasi deb, ular ayirmasining tur o‘lcha-miga bo‘linishiga aytiladi. Ko‘rsatkichlarni o‘zaro qo‘shish mumkin emas.