

# Програмчлалын C++ хэл дээрх санах ой, хаяг, хаяган хувьсагч, заалт болон санах ойн цоорхой зэрэг зүйлсийг судлан, асуултуудад хариулж, код бичих лабораторийн ажил

(Лаборатори №2)

Э.Түвшинтөгс | 20B1NUM0448

МУИС - ХШУИС, Компьютерийн Ухаан, 3-р түвшин, [tuvshintugs@mail.com](mailto:tuvshintugs@mail.com)

## 1. ОРШИЛ

Програмчлалын C++ хэл дээрх санах ой болон түүний хаяг, хаяган хувьсагч, заалт, санах ойн цоорхой зэрэг зүйлсийг судлан түүнийгээ ашиглан лабораторийн ажил дээр өгөгдсөн асуултад хариулан, шинжлэнэ.

## 2. ЗОРИЛГО

Санах ойн хувьсагч болон түүний хаяг, цаашлаад хаяган хувьсагч, заалт зэрэг зүйлсийн тухай ойлголттой болон мэдэж авсан ойлголтуудаа ашиглан хувьсагчуудын утгыг солих функцийг бичиж сурах, ажиллах зорилготой.

## 3. ОНОЛЫН СУДАЛГАА

### 3.1 Санах ойн хаяг гэж юу вэ?

Компьютерын санах ой дээр утга болгон өөр өөрийн гэсэн хаягтай байдаг. Харин тэдгээрлүү “&” ашиглан хандах боломжтой.

Жишээ нь:

```
int x = 5;  
int *test = &x;
```

гэвэл test-ийн утгад x-ийн хаягийг оноож өгч байна гэсэн үг. Өөрөөр хэлбэл x-ийг хадгалж буй санах ойн хаяг.

### 3.2 Хаяган хувьсагч гэж юу вэ? Хаяган хувьсагчийн төрөл ямар байдаг мөн багтаамж нь хэдэн байт байдаг вэ?

Хаяган хувьсагч гэдэг нь санах ойн хаягийг хадгалдаг хувьсагч юм. Хаяган хувьсагч нь pointer бөгөөд \* ашиглан зарлаж бас тухайн утгаруу ханддаг.

Багтаамжийн хувьд хаяган хувьсагч нь тухайн үйлдлийн системээс хамаардаг. Хэрэв 64 бит үйлдлийн системтэй бол 8 байт, 32 бит үйлдлийн системтэй бол 4 байт байна.

### 3.3 New operator гэж юу вэ? Ямар үйлдэл хийж юу буцаадаг вэ?

“new” operator нь сул санах ойгоос санах ой нөөцлөхөд хэрэглэгддэг. New operator-ийг дуудах үед сул санах ойн хэсгээс санах ой нөөцлөөд, тухайн програмд оноож өгөн түүний эхлэл хаягийг буцаадаг.

Програм ажилууллахаасаа өмнө эхлээд буцаах хаягийг шалган, алдаа зааж магадгүй буюу сул санах ой байгаа эсэхийг шалгах хэрэгтэй.

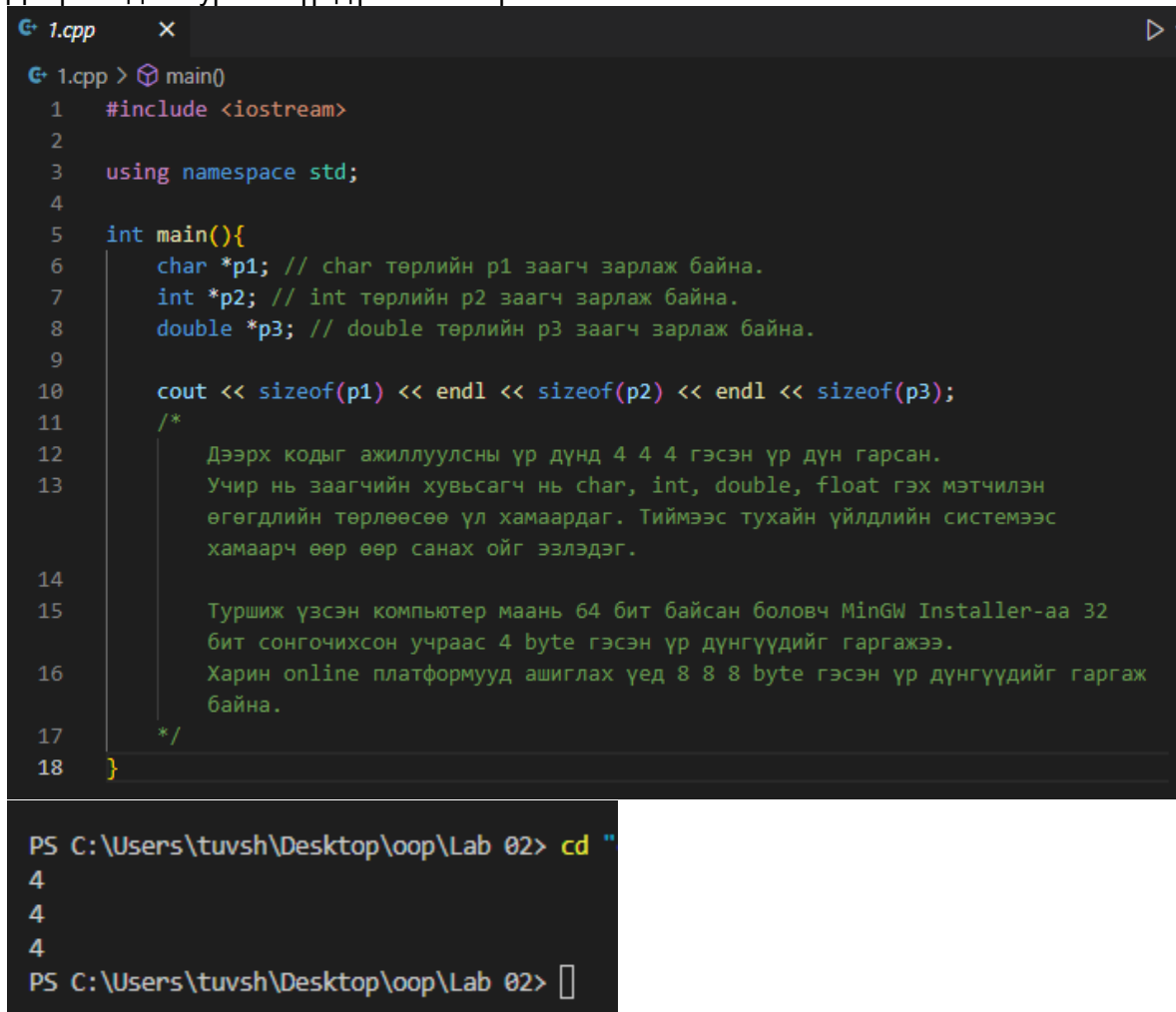
### 3.4 Санах ойн цоорхойн талаар дэлгэрэнгүй тайлбарла. Ойн цоорхой гэж юу болох, ямар тохиолдолд үүсэх, яаж сэргийлэх гэх мэт.

Санах ойн цоорхой гэдэг нь санах ой дээр нөөцлөсөн, ахин ашиглах боломжгүй болгосон санах ойг хэлнэ.

new operator ашиглаж байгаа үед санах ой чөлөөлөгддөггүй. Тиймээс delete operator ашиглан тухайн нөөцлөгдөж, “цоорхой” болж үлдсэн санах ойг чөлөөлөх хэрэгтэй.

## 4. ХЭРЭГЖҮҮЛЭЛТ БОЛОН ҮР ДҮН

- 1) Доорх кодыг туршиж үр дүнг тайлбарлан бич.



```
1.cpp
1.cpp > main()
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      char *p1; // char төрлийн p1 заагч зарлаж байна.
7      int *p2; // int төрлийн p2 заагч зарлаж байна.
8      double *p3; // double төрлийн p3 заагч зарлаж байна.
9
10     cout << sizeof(p1) << endl << sizeof(p2) << endl << sizeof(p3);
11     /*
12      Дээрх кодыг ажиллуулсны үр дүнд 4 4 4 гэсэн үр дүн гарсан.
13      Учир нь заагчийн хувьсагч нь char, int, double, float гэх мэтчилэн
14      өгөгдлийн төрлөөсөө үл хамаардаг. Тиймээс тухайн үйлдлийн системээс
15      хамаарч өөр өөр санах ойг эзлэдэг.
16
17      Туршиж үзсэн компьютер маань 64 бит байсан боловч MinGW Installer-аа 32
18      бит сонгочихсон учраас 4 byte гэсэн үр дүнгүүдийг гаргажээ.
19      Харин online платформууд ашиглах үед 8 8 8 byte гэсэн үр дүнгүүдийг гаргаж
20      байна.
21     */
22 }
```

```
PS C:\Users\tuvsh\Desktop\oop\Lab 02> cd ""
4
4
4
PS C:\Users\tuvsh\Desktop\oop\Lab 02> 
```

- 2) Доорх кодыг туршиж мөр бүрийн үр дүнг тайлбарлан бич.

```
2.cpp X
G 2.cpp > main()
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int a = 125; // Бүхэл тоог хувьсагч зарлаж, 125 гэсэн утга өгч байна.
7      int *p = &a; // Pointer төрлийн p хувьсагч зарлан, a хувьсагчийн хаягийг өгч
        байна.
8
9      cout<<p<<endl; // p хаяган хувьсагчийг буюу a-ийн хаягийг хэвлэж байна.
10     cout<<*p<<endl; // p хаяган хувьсагчид хадгалагдаж байгаа a хувьсагчийг
        хаягаар нь дамжуулан хэвлэж байна.
11
12     p++; // p хаяган хувьсагчийг нэгээр нэмэгдүүлэх буюу дараачийн 4 byte хаягруу
        нэмэн, шилжүүлж байна.
13     cout<<p<<endl; // 4 байтаар нэмэгдэн, шилжсэн p-ийн хаягийн утгыг хэвлэж байна.
14     cout<<*p<<endl; // p хаяган хувьсагчид байгаа утгад оноогдсон утгыг хэвлэж
        байна.
15 }
```

```
PS C:\Users\tuvsh\Desktop\oop\Lab 02> cd
0x61ff08
125
0x61ff0c
6422284
PS C:\Users\tuvsh\Desktop\oop\Lab 02> 
```

3) Доорх кодын мөр бүрийг тайлбарла.

```

3.cpp > main()
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int numbers[5]; // 5 тоо авах боломжтой numbers гэх хүснэгт зарлаж байна.
7      int *p; // p гэх pointer утга зарлаж байна.
8
9      p = numbers; *p = 10; // p-ийн утгад numbers хүснэгтийг оноож өгөн, эхний *p
      заагч 4 byte-д 10 гэсэн утгыг өгч байна.
10
11     p++; *p = 20; // p заагчийг нэг нэгээр нэмэгдүүлэх буюу дараачийн 4 байт
      хаяганд зааж өгч байна. Мөн тухайн зааж өгсөн дараагийн 4 байт хаяганд 20
      гэсэн утгыг өгч байна.
12
13     p = &numbers[2]; *p = 30; // p-ийн утгад numbers хүснэгтийн 3 дахь утгыг зааж
      өгнө. Дараа нь тэр заасан хувьсагчдаа 30-ийг оноож өгсөн.
14
15     p = numbers + 3; *p = 40; // p-ийн утгыг 3 аар нэмэгдүүлж, 4 дэх элементрүү
      шилжүүлж байна. Дараа нь тэр утгад 40-ийг оноож өгч байна.
16
17     p = numbers; *(p+4) = 50; // p-д numbers буюу хүснэгтийн хамгийн эхний
      хаягийг зааж өгч байна. Дараа нь хүснэгтийн 4 дэх элементийн хаягруу 50 гэсэн
      утга өгч байна. Өөрөөр хэлбэл numbers[4] = 50 гэсэн үг.
18
19
20     for (int n=0; n<5; n++)
21         cout << numbers[n] << ", ";
22     /*
23     numbers хүснэгтэд хадгалагдсан тоонуудыг нөхцөлд давталт ашиглан хэвлэж
      байна. Мөн хэвлэхдээ ард нь таслалтайгаар хэвлэж байна.
24     */
25 }

PS C:\Users\tuvsh\Desktop\oop\Lab 02> cd
10, 20, 30, 40, 50,
PS C:\Users\tuvsh\Desktop\oop\Lab 02>

```

- 4) Хаяган хувьсагч ашиглан (функцин параметр нь хаяган хувьсагч байна) хоёр хувьсагчийн утгыг солих хэрэглэгчийн функц бич.

```
4.cpp  X  ▶
4.cpp > main()
1  #include <iostream>
2
3  // Хаяган хувьсагч ашиглан (функцын параметер нь хаяган хувьсагч байна) хоёр
   хувьсагчийн утгыг солих хэрэглэгчийн функц бич.
4
5  using namespace std;
6
7  void swap(int *a, int *b){ // swap гэх нэртэй a болон b гэх утгуудыг авдаг хаяган
   хувьсагчид зарлана.
8      int temp = *a; // a-ийн утгыг түр хадгалах temp хувьсагчид оноож өгнө.
9      *a = *b; // Дараа нь b-ийн утгыг a-д хадгална.
10     *b = temp; // Дараа нь өмнө нь түр утгад хадгалсан a-ийнхаа утгыг b-д
        хадгалснаар 2 тоог солих үйлдлийг амжилттай гүйцэтгэнэ.
11 }
12
13 int main(){
14     int a = 5, b = 10; // a болон b хувьсагчдыг зарлан, утга оноож өгч байна.
15     cout << a << " " << b << endl; // 2 хувьсагчийн утгуудын байрыг солихоос өмнө
16
17     swap(&a, &b); // Байрыг нь солих функцаа дуудан ажилуулж байна.
18     cout << a << " " << b; // Хувьсагчуудын байрыг сольсны дараа
19
20     return 0;
21 }
```

```
PS C:\Users\tuvsh\Desktop\oop\Lab 02>
4 }
5 10
10 5
```

- 5) Заалтан хувьсагч (функцын параметер нь заалт байна) хоёр хувьсагчийн утгыг солих хэрэглэгчийн функц бич.

```
5.cpp > swap(int &, int &)
1  #include <iostream>
2
3  // Заалтан хувьсагч (функцын параметер нь заалт байна) хоёр хувьсагчийн утгыг
   солих хэрэглэгчийн функц бич.
4
5
6  using namespace std;
7
8  void swap(int &a, int &b){ // Параметр нь заалт байх swap функцийг үүсгэнэ.
9      int temp = a; // а-ийн утгыг түр хадгалах temp хувьсагчид оноож өгнө.
10     a = b; // Дараа нь b-ийн утгыг а-д хадгална.
11     b = temp; // Дараа нь өмнө нь түр утгад хадгалсан а-ийнхаа утгыг b-д
        хадгалснаар 2 тоог солих үйлдлийг амжилттай гүйцэтгэнэ.
12
13 }
14
15 int main(){
16     int a = 5, b = 10; // а болон b хувьсагчдыг зарлан, утга оноож өгч байна.
17     cout << a << " " << b << endl; // 2 хувьсагчийн утгуудын байрыг солихоос өмнө
18
19     swap(a, b); // Байрыг нь солих функцаа дуудан ажилуулж байна.
20     cout << a << " " << b; // Хувьсагчуудын байрыг сольсны дараа
21
22     return 0;
23 }
```

```
PS C:\Users\tuvsh\Desktop\oop\Lab 02>
5 }
5 10
10 5
```

## 5. ДҮГНЭЛТ

Энэхүү лабораторийн үр дүнд санах ойн талаар мэдлэгийг хуримтлууллаа. Ийнхүү цаашдаа санах ой, заалт, хаяг гэх мэтчилэн сэдвүүдийн хүрээнд ажиллах боломжтой боллоо. Мөн санах ойн цоорхой, түүнийг чөлөөлөх зэрэг зүйлсийн талаар мэдлэгтэй болсноор програм ажиллах үед гарч болох удаашрал, гацалт, их ачаалал үүсч үйлдлийн системийн үйлдэл удаашрах гэх мэт зүйлсээс сэргийлэхийг сурлаа. Мөн заалт ашиглан санах ой хэмнэх боломжтой гэдгийг мэдлээ.

## 6. АШИГЛАСАН МАТЕРИАЛ

<https://www.geeksforgeeks.org/new-and-delete-operators-in-cpp-for-dynamic-memory/>

<https://www.geeksforgeeks.org/c-pointers/?ref=lbp>

<https://www.geeksforgeeks.org/references-in-c/?ref=lbp>

## 7. ХАВСРАЛТ

```
18 lines (14 sloc) | 1.15 KB
Raw Blame

1 #include <iostream>
2
3 using namespace std;
4
5 int main(){
6     char *p1; // char төрлийн p1 заагч зарлаж байна.
7     int *p2; // int төрлийн p2 заагч зарлаж байна.
8     double *p3; // double төрлийн p3 заагч зарлаж байна.
9
10    cout << sizeof(p1) << endl << sizeof(p2) << endl << sizeof(p3);
11    /*
12     Дээрх кодыг ажиллуулсны үр дүнд 4 4 4 гэсэн үр дүн гарсан.
13     Учир нь заагчийн хувьсагч нь char, int, double, float гэх мэтчилэн өгөгдлийн төрлөөсөө үл хамаардаг. Тиймээс тухайн үйлдлийн системээс хамаарч өөр өөр санах ойг эзлэдэг.
14
15     Туршиж үзсэн компьютер маань 64 бит байсан боловч MinGW Installer-аа 32 бит сонгочихсон учраас 4 byte гэсэн үр дүнгүүдийг гаргажээ.
16     Харин online платформууд ашиглах үед 8 8 8 byte гэсэн үр дүнгүүдийг гаргаж байна.
17    */
18 }
```

```
15 lines (11 sloc) | 1.05 KB
Raw Blame

1 #include <iostream>
2
3 using namespace std;
4
5 int main(){
6     int a = 125; // Бүхэл тоог хувьсагч зарлаж, 125 гэсэн утга өгч байна.
7     int *p = &a; // Pointer төрлийн p хувьсагч зарлан, a хувьсагчийн хаягийг өгч байна.
8
9     cout<<p<<endl; // p хаяган хувьсагчийг буюу a-ийн хаягийг хэвлэж байна.
10    cout<<*p<<endl; // p хаяган хувьсагчид хадгалагдаж байгаа a хувьсагчийг хаягаар нь дамжуулан хэвлэж байна.
11
12    p++; // p хаяган хувьсагчийг нэгээр нэмэгдүүлэх буюу дараачийн 4 byte хаягруу нэмэн, шилжүүлж байна.
13    cout<<p<<endl; // 4 байтаар нэмэгдэн, шилжсэн p-ийн хаягийн утгыг хэвлэж байна.
14    cout<<*p<<endl; // p хаяган хувьсагчид байгаа утгад оноогдсон утгыг хэвлэж байна.
15 }
```

```
25 lines (16 sloc) | 1.78 KB
Raw Blame

1 #include <iostream>
2
3 using namespace std;
4
5 int main(){
6     int numbers[5]; // 5 тоо авах боломжтой numbers гэх хүснэгт зарлаж байна.
7     int *p; // p гэх pointer утга зарлаж байна.
8
9     p = numbers; *p = 10; // p-ийн утгад numbers хүснэгтийг оноож өгөн, эхний *p заагч 4 byte-д 10 гэсэн утгыг өгч байна.
10
11    p++; *p = 20; // p заагчийг нэг нэгээр нэмэгдүүлэх буюу дараачийн 4 байт хаяганд зааж өгч байна. Мөн тухайн зааж өгсөн дараагийн 4 байт хаяганд 20 гэсэн утгыг өгч байна.
12
13    p = &numbers[2]; *p = 30; // p-ийн утгад numbers хүснэгтийн 3 дахь утгыг зааж өгнө. Дараа нь тэр заасан хувьсагчдаа 30-ийг оноож өгсөн.
14
15    p = numbers + 3; *p = 40; // p-ийн утгыг 3 аар нэмэгдүүлж, 4 дэх элементрүү шилжүүлж байна. Дараа нь тэр утгад 40-ийг оноож өгч байна.
16
17    p = numbers; *(p+4) = 50; // p-д numbers буюу хүснэгтийн хамгийн эхний хаягийг зааж өгч байна. Дараа нь хүснэгтийн 4 дэх элементийн хаягруу 50 гэсэн утга өгч байна. Өөрөөр хэлбэл 5-ийн индексийн хаяганд 50-ийг оруулж байна.
18
19
20    for (int n=0; n<5; n++)
21        cout << numbers[n] << ", ";
22    /*
23     numbers хүснэгтэд хадгалагдсан тоонуудыг нөхцөлд давталт ашиглан хэвлэж байна. Мөн хэвлэхдээ ард нь таслалтайгаар хэвлэж байна.
24    */
25 }
```

21 lines (15 sloc) | 1.3 KB

Raw

Blame



```
1 #include <iostream>
2
3 // Хаяган хувьсагч ашиглан (функцин параметер нь хаяган хувьсагч байна) хоёр хувьсагчийн утгыг солих хэрэглэгчийн функц бич.
4
5 using namespace std;
6
7 void swap(int *a, int *b){ // swap гэх нэртэй a болон b гэх утгуудыг авдаг хаяган хувьсагчид зарлана.
8     int temp = *a; // a-ийн утгыг түр хадгалах temp хувьсагчид оноож өгнө.
9     *a = *b; // Дараа нь b-ийн утгыг a-д хадгална.
10    *b = temp; // Дараа нь өмнө нь түр утгад хадгалсан a-ийнхаа утгыг b-д хадгалснаар 2 тоог солих үйлдлийг амжилттай гүйцэтгэнэ.
11 }
12
13 int main(){
14     int a = 5, b = 10; // a болон b хувьсагчдыг зарлан, утга оноож өгч байна.
15     cout << a << " " << b << endl; // 2 хувьсагчийн утгуудын байрыг солихоос өмнө
16
17     swap(&a, &b); // Байрыг нь солих функцаа дуудан ажилуулж байна.
18     cout << a << " " << b; // Хувьсагчуудын байрыг сольсны дараа
19
20     return 0;
21 }
```

23 lines (15 sloc) | 1.22 KB

Raw

Blame



```
1 #include <iostream>
2
3 // Заалтан хувьсагч (функцин параметер нь заалт байна) хоёр хувьсагчийн утгыг солих хэрэглэгчийн функц бич.
4
5
6 using namespace std;
7
8 void swap(int &a, int &b){ // Параметр нь заалт байх swap функцийг үүсгэнэ.
9     int temp = a; // a-ийн утгыг түр хадгалах temp хувьсагчид оноож өгнө.
10    a = b; // Дараа нь b-ийн утгыг a-д хадгална.
11    b = temp; // Дараа нь өмнө нь түр утгад хадгалсан a-ийнхаа утгыг b-д хадгалснаар 2 тоог солих үйлдлийг амжилттай гүйцэтгэнэ.
12
13 }
14
15 int main(){
16     int a = 5, b = 10; // a болон b хувьсагчдыг зарлан, утга оноож өгч байна.
17     cout << a << " " << b << endl; // 2 хувьсагчийн утгуудын байрыг солихоос өмнө
18
19     swap(a, b); // Байрыг нь солих функцаа дуудан ажилуулж байна.
20     cout << a << " " << b; // Хувьсагчуудын байрыг сольсны дараа
21
22     return 0;
23 }
```