# C++ хаяган болон заалтан хувьсагч, дотоод мөр функцийн хэрэглээг судлах лабораторийн ажил (Лаборатори №2)

Я.Тэмүүжин

ХШУИС, МКУТ, ПХ 16B1SEAS3369, y.usott@gmail.com

# 1. ОРШИЛ

С++ хэлний санах ойн хаяг, ойн цоорхой, хаяган хувьсагч, заалтан хувьсагчийг судалж ойлгосны үндсэн дээр хавсралтанд өгөгдсөн кодыг судалж турших, тайлбарлах, бодлого бодож заалтан хувьсагч болон мөр функцын хэрэглэх давуу тал, дутагдалтай талыг тайлбарлан хэрэгжүүлэх болно.

# 2. ЗОРИЛГО

Санах ойн хаяг, хаяган хувьсагч, заалт, дотоод мөр функц, ойн цоорхойн талаар судалж, хэрэглээнд оруулж туршин ажиглалт явуулж тайлбарлана. Хаяган болон заалтан хувьсагч ашиглан хоёр хувьсагчийн утгийг солих хэрэглэгчийн функц бичнэ. Үүний тулд дараах зорилтуудыг тавьж ажиллана.

1. Хувьсагчдыг параметрээр илгээхэд доторх утгыг солих хэрэглэгчийн функц зарлаж, тодорхойлно
2. Хувьсагчдыг параметрээр илгээхэд доторх утгыг хэвлэх хэрэглэгчийн функц зарлаж, тодорхойлно
3. Хэрэглэгчийн функцыг дуудахдаа хувьсагчийн хаягийг дамжуулна.
4. Хэрэглэгчийн функцийг дуудхад дамжуулах хувьсагчийн заалтыг функцийн параметрт бэлдэж функц дотроо заалтан хувьсагчаа ашиглана.
5. Утга солих үйлдэл хийхийн өмнө болон дараа хувьсагчдийн дэлгэцэнд хэвлэж үйлдэл зөв хийгдэж байгаа эсэхийг шалгана.

# 3. ОНОЛЫН СУДАЛГАА

## Санах ойн хаяг

Орчин үеийн компьютер нь байт-хаяглалт ашигладаг ба хаяг бүр нь байт(8 бит) зайг заадаг. [1] Санах ойн хаяг нь CPUд санах ойн байрлал бүрийг хянахад зориулж санах ойн үүр бүрт өгөгдсөн дахин давтагдахгүй тоон утга байдаг [2]. Өөрөөр хэлбэл хаяг бүрийн ард нэг санах ойн үүр байдаг ба санах ойн үүр тухайн системээс хамаарч ихэвчлэн 4 эсвэл 8 байт хэмжээтэй байдаг. Санах ойн хаягийн хэмжээний хувьд жишээ нь 32 битийн компьютер 32 битийн санах ойн хаягийг зөвшөөрдөг бөгөөд 232= 4,294,967,296 байт өөрөөр 4гегабайт санах ойг хаяглах боломжтой. [1]

## Хаяган хувьсагч

Хаяг бол тоон утга тул түүнийг хувьсагч руу хадгалж болох ба хаяг хадгалах хувьсагчийг хаяган хувьсагч гэнэ.

32 битийн машин болон үйлдлийн системд хаяган хувьсагч нь санах ойд 4 байт хэмжээ эзэлдэг бол 64 битийн машин болон үйлдлийн системд хяган хувьсагч нь 8 байт хэмжээ эзэлдэг. Мөн 64 битийн машин 32 битийн үйлдлийн систем суусан бол 4 байт хэмжээ эзэлнэ.

## Заалтан хувьсагч

Заалтан хувьсагч нь өмнө нь зарлагдсан хувьсагчийн өөр нэг нэр гэж хэлж болно. Зарлахдаа заавал гарааны утга өгөх ёстой. Гарааны утга нь өмнө нь зарлагдсан хувьсагч байна. [2]

Жишээ нь: Доорх кодыг ажиглавал b хувьсчийн заалтан хувьсагч ба өөрөөр хэлвэл b нь a хувьсагчийн өөр нэг нэр гэж хэлж болно. Яг л Батаа, Дэлгэр гэдэг хочтой хүчнийг Батаа гэж дуудсан Дэлгэр гэж дуудсан тухайн хүн ялгай байхгүй шиг.

1. Int a = 5;
2. Int &b = a;

## Дотоод мөр функц

С++ хэлэнд функц болон програмын хурдыг нэмэгдүүлэх зорилгоор мотоодөр функц буюу inline function ашиглах боломж нэмэгдсэн. [2] Си хэлэн функц дуудах үе бүрт дараах үйлдэл хийгддэг

* Дараагийн ээлжинд хийгдэх командын хаягийг хадгалах
* CPU дотоод регистрийн утгыг хадгалах
* Дуудагдсан функц рүү дамжуулж өгч байгаа аргументийг стек рүү хадгалах
* Програмын удирдлага дуудагдсан функц рүү шилжин очиж улмаар аргументээ стекээс авах
* Функц ажиллаж дуусахад түүнээс буцах утгыг үндсэн програм, эх функц нь гаргаж авахаар CPU регистр эсвэл стек рүү түр хадгалах
* Програмын удирдлага эх функц рүү шилжихдээ өмнө хадгалсан буцах хаяг, CPU регистрийн утгыг стекээс буцааж гаргах [3]

Функц хэрэглэх үед дээрх бүх үйлдэлд нэмэлт CPU цаг шаардагдана. Ийм учраас функц болон програмын хурд удааширдаг байна. Харин дотоод мөр функцийг ашиглаж функц ашиглах бүртээ командуудыг функцийг хэрэглэсэн командын мөр бүрд хувилж үүсгэнэ. Ингэснээс өмнө нь харуулсан хэрэглэгчийн функц дуудхад хийхэд хийгддэг үйлдлүүд хийгдэхгүй ба програм болон функцийн хурд нэмэгдэнэ. Харин сул тал нь програмын хэмжээ ихсэх бичиглэл нэмэгдэх талтай.

Дотоод мөр функц тодорхойлох

1. inline int name() {
3. }

Дотоод мөр функц дуудах

1. name();

өрөөр хэлбэл програм compiler кодыг компайл хийхдээ функцийн нэрний оронд түүний кодыг хуулж тавьдаг гэсэн үг юм.

## Ойн цоорхой

Ойн цоорхой (memory leaks) бол new, delete операторуудыг дэс дараатайгаар олон удаа хэрэглэх програмд гарч болох ойлголт юм. Өөрөөр хэлбэл new операторыг ашиглаж санах ой нөөцлөөд ашиглаж дуусасны дараа delete оператороор чөлөөлөх үйлдэл хийгээгүйгээс болж үүсдэг [3]. Чөлөөлөх үйлдэл хийгдээгүй санах ойн хэсэгт хандах боломжгүй тул санах энэ үйлдэл олон давтагдвал санах ой дүүрэх эрсдэлтэй.

Жишээ нь доорх функц нь санах ой нөөцлөн ашиглаж байгаа боловч ашиглаж дуусасны дараа ойгоо чөлөөлөхгүй байгаан улмаас санах ойн цоорхой үүсгэж байгаа байна.Дараах функцыг олон удаа дуудхад санах ой дүүрнэ.

1. void function\_which\_allocates(void) {
2. float \* a = new int[45];
3. }
4. int main(void) {
5. function\_which\_allocates();
6. }

# 4. ХЭРЭГЖҮҮЛЭЛТ

## 4.1. - Хаяган хувьсагч ашиглан (функцын параметер нь хаяган хувьсагч байна) хоёр хувьсагчийн утгыг солих хэрэглэгчийн функц бич.

Тус бодлогыг бодохдоо хаяган хувьсагч ашиглаж байгаа тул утга солих болон хэвлэх функцийн төрлийг утга буцаахгүй буюу void төрлөөр зарлаж тодорхойлсон. Хэрэглэгчээс бүхэл тоон хоёр тоо авч тус тусад нь хувьсагчид хадгалсан. Үүний дараа утга солих хэрэглэгчийн функцрүү тэдгээрийн хаягийг хаягийн оператор ашиглаж параметрээр дамжуулсан. Хэрэглэгчийн утга солих функцэд хаяган хувьсагч болон нэмэлт хувьсагчийн тусламжтайгаар хоёр хаяганд байгаа утгыг солисон. Кодыг хавсралт 7.4-с харна уу.

## 4.2. - Заалтан параметр (функцын параметер нь заалт байна) ашиглан хоёр хувьсагчийн байрыг солих хэрэглэгчийн функц бич.

Бодлогыг бодохдоо main функцэд утга солих хувьсагчийн зарлаж утга оноож байгаа мөн хэрэглэгчийн функц заалт ашиглаж байгаа тул утга солих болон хэвлэх функцийн төрлийг утга буцаахгүй буюу void төрлөөр зарлаж тодорхойлсон. Хэрэглэгчээс бүхэл тоон хоёр тоо авч тус тусад нь хувьсагчид хадгалсан. Үүний дараа утга солих хэрэглэгчийн функцрүү тэдгээрийн параметрээр дамжуулсан. Хэрэглэгчийн утга солих функцэд заалт болон нэмэлт хувьсагчийн тусламжтайгаар хоёр хаяганд байгаа утгыг солисон.Кодыг хавсралт 7.5-с харна уу.

# 5. ДҮГНЭЛТ

Санах ойн үүр бүрт хаяг байдаг ба тухайн систем хэдэн битийнхээс хамаарч хэмжээ нь өөр өөр байдаг жишээ нь 32бит бол 4байт 64бит бол 8 байт хэмжээтэй ба үүнийг санах ойн хаяг гэнэ. Хаяган хувьсагч нь өөрийн хаягтай хувьсагч бөгөөд утга дээрээ зөвхөн хаяг хадгалдгаараа онцлог. Харин заалт нь өмнө зарлагдсан хувьсагчид өөр нэр өгөж ашиглахад ашигладаг ба зарлагдахдаа заавал гарааний утгатай байх ёстой өөрийн хаяггүй шууд гарааны утгийн хаягруу заадаг нэр юм. Санах ой нөөцөлж түүнийг ашиглаж дуусасны дараа чөлөөлөх үйлдэл хийхгүйгээр орхих нь санах ойн цоорхойг үүсгэдэг.

6. АШИГЛАСАН МАТЕРИАЛ

|  |  |
| --- | --- |
| [1] | “What is a Memory Address? - Definition from Techopedia,” [Холбогдсон]. Available: https://www.techopedia.com/definition/323/memory-address. |
| [2] | “Understand C++ function with examples,” 09 09 2018. [Холбогдсон]. Available: https://www.programiz.com/cpp-programming/function. |
| [3] | Ж.Пүрэв, Объект хандлагат технологийн С++ програмчлал, Улаанбаатар, 2008. |
| [4] | “https://en.wikipedia.org/wiki/Memory\_address,” [Холбогдсон]. Available: https://en.wikipedia.org/wiki/Memory\_address. [Нэвтэрсэн 12 09 2018]. |

# 7. ХАВСРАЛТ

## 7.1 Код 1 - Доорх кодыг туршиж үр дүнг шинжлэн тайлабарлан бич.

Доорх код нь гурван өөр төрлийн заагч хувьсагчыг зарлаж sizeof() функц ашиглан тус бүр санах ойд хэдэн байт хэмжээ эзлэж байгааг хэвлэж байна. Санах ойн хаягийн хэмжээ тухайн үйлдлийн системээс хамаарч өөр өөр байдаг 32 битийн системийн санах ойн хаяг 4 байт байдаг бол 64 бит үйлдлийн системд 8 байт байдаг. Доорх жишээг 64 битийн үйлдлийн систем дээр ажлуулсан тул 8байт хэмжээтэй гэж хэвлэсэн.

1. #include<iostream>
2. using namespace std;
3. int main() {
4. //доорх зарлалт нь p1 заагчид байрлах санах ойн хаягд дахь утгийг char төрлөөр илэрхийлэнэ гэсэн утгатай.
5. char \* p1;
6. int \* p2;
7. double \* p3;
8. cout << sizeof(p1) << sizeof(p2) << sizeof(p3);
9. }

Үр дүн

888

Бүгд ижил гарч байгаа учир нь санах ойн хаяг тогтсон урттай тоон хэмжээсээр илэгхийлэгддэг учраас төрөл харгалзахгүй ижил хэмжээтэй байна харин заагч зарлахдаа төрөл ашигладаг нь тухайн хаяг дахь санах ойн үүрийг ямар төрлөөр илэрхийлэхийг заадаг байна.

## 7.2 Код 2 - Доорх кодыг туршиж мөр бүрийн үр дүнг шинжлэн тайлбарла.

1. // Бүхэл тоон a хувьсагч зарлаж түүнд 125 утгыг оноож байна
2. int a = 125;
3. // Бүхэл тоон төрлийн p заагчин хувьсагч зарлаж түүнд a хувьсагчийн хаягийг хаягийн оператор &-н тусламжтайгаар оноож байна
4. int \*p = &a;
5. // p заагчин хувьсагчийн агуулж буй хаягийг хэвлэж байна
6. cout<<p;
7. // p заагчин хувьсагчийн агуулж буй хаяг доторх утгийг хэвлэж байна
8. cout<<\*p;
9. //p-г дараагийн санах ойд шилжүүлнэ. Ингэхдээ p-г илэрхийлж буй өгөгдлийн төрлийн санах ойд эзлэх хэмжээг p-н хэмжээ дээр нэмнэ. Энэ тохиолдолд int нь 4 байт тул хаяг 180 байсан гэхэд 184 дээр байна гэсэн үг
10. p++;
11. // p-н өөрчлөгдсөн хаягийг хэвлэж байна
12. cout<<p;
13. // p-н өөрчлөгдсөн хаяг дээрх утгийг хэвлэж байна.
14. // энэ хаягт утга оноогоогүй учир санах ойн хог хэвлэнэ.
15. cout<<\*p;

## 7.3 Код 3

1. //зургаан элементтэй бүхэл тоон төрлийн numbers хувьсагч үүсгэж байна
2. int numbers[5];
3. // бүхэл тоон төрлийн p заагч хувьсагч зарлаж байна
4. int \* p;
5. //p-д numbers массивын эхний элементийн хаягийг олгоод тус хаяг дотрох утган 10 –г оноож байна.
6. p = numbers; \* p = 10;
7. //p доторх хаягийг 4 байтаар нэмэгдүүлж(int төрлийн массив тул 1-р нэмэгдүүлнэ гэдэг нь numbers хүснэгтийн төрөл буюу int –н санах ойд байрлах хэмжээ буюу 4 байтыг одоо байгаа хаяган дээр нэмнэ гэсэн үг) тус нэмэгдүүлсэн хаягийн доторох утганд 20 –г оноож байна.
8. p++; \* p = 20;
9. // p-д numbers хүснэгтийн 2-р индексийн хаягийг олгоод. Тус хаяг доторх утганд 30-г олгож байна.
10. p = & numbers[2]; \* p = 30;
11. // p-д numbers хүснэгтийн эхний элементийн хаяг дээр 3-г нэмээд тус хаягийг олгоод p доторх хаяг дээрх утганд 40-г олгоэо.
12. p = numbers + 3; \* p = 40;
13. //p-д numbers хүснэгтийн эхний хаягийг зааж өгөөд p + 4 буюу эхний хаяг дээр 4-г нэмээд гарсан хаягийг утгад 50-г олгож байна
14. p = numbers; \* (p + 4) = 50;
15. //Давталтын тоолуурыг 0 болгоод n<5 нөхцөлд үнэн үед биелэнэ.
16. for (int n = 0; n < 5; n++) {
17. //numbers –н n-р элементийг хэвлэж байна
18. cout << numbers[n] << ", ";
19. }

## 7.4 Бодлого 1 - Хаяган хувьсагч ашиглан (функцын параметер нь хаяган хувьсагч байна) хоёр хувьсагчийн утгыг солих хэрэглэгчийн функц бич.

1. #include<iostream>
2. using namespace std;
3. //дамжуулсан хоёр хаягийн утгийг солих функц зарлаж байна
4. void swap(int \*x , int \*y );
5. //дамжуулсан хоёр хаягийн утгийг хэвлэх функц зарлаж байна
6. void display(int \* , int \* );
7. int main() {
8. //хэрэглэгч оруулсан хоёр тоог хадгалах хувьсагч
9. int x, y;
10. //хэрэглэгчид зориулж заавар гаргаж байна
11. cout<<”X,Y=”;
12. //Хэрэглэгч утга оруулж байна
13. cin >> x >> y;
14. // Хэрэглэгчийн оруулсан утгуудыг хэвлэж байна
15. display(&x,&y);
16. //Хэрэглэгчийн оруулсан хоёр тоог хадгалж буй хувьсагчдийн хаягийг дамжуулж утгийг нь сольж байна
17. swap(&x, &y);
18. //хоёр хувьсагчийн утга өөрчлөгдсөн эсэхийг хэвлэж шалгаж байна
19. display(&x, &y);
20. }
22. /\*дамжуулсан хоёр хаягийн утгийг солих функц зарлаж байна
23. x- дамжуулсан х хаягийг хадгалах хаяган хувьсагч
24. y- дамжуулсан y хаягийг хадгалах хаяган хувьсагч
25. \*/
26. void swap(int \* x, int \* y) {
27. //int төрлийн temp хувьсагчийг зарлаж x хаяган хувьсагчийн хадгалж буй хаяг дээр буй утгийг хадгалж авч байна.
28. int temp = \* x;
29. //x-д буй хаяган дахь утганд y-д байгаа хаяган дахь утгийг оноож байна
30. \*x = \*y;
31. //x-д буй хаяган дахь утганд temp-д байгаа утга буюу y-д байсан хаяган дахь утгийг оноож байна
32. \*y = temp;
33. }
34. /\*
35. дамжуулсан хоёр хаягийн утгийг хэвлэх функц зарлаж байна
36. x- дамжуулсан х хаягийг хадгалах хаяган хувьсагч
37. y- дамжуулсан y хаягийг хадгалах хаяган хувьсагч
38. \*/
39. void display(int \* x, int \* y) {
40. //параметрээр илгээсэн хоёр хаяган хувьсагчийн утгийг cout объектийн << операторыг ашиглаж хэвлэж байна.
41. cout << "x= " << \* x << " y=" << \* y << endl;
42. }

## 7.5 Бодлого 2 - Заалтан параметр (функцын параметер нь заалт байна) ашиглан хоёр хувьсагчийн байрыг солих хэрэглэгчийн функц бич.

1. #include<iostream>
2. using namespace std;
3. /\*
4. дамжуулсан хоёр хувьсагчийн утгийг солих функц зарлаж байна
5. \*/
6. void swap(int & , int & );
7. //дамжуулсан хоёр хувьсагчийн утгийг хэвлэх функц зарлаж байна
8. void display(int & , int & );
9. int main() {
10. //хэрэглэгч оруулсан хоёр тоог хадгалах хувьсагч
11. int x, y;
12. //хэрэглэгчид зориулж заавар гаргаж байна
13. cout<<”X,Y=”;
14. //Хэрэглэгч утга оруулж байна
15. cin >> x >> y;
16. // Хэрэглэгчийн оруулсан утгуудыг хэвлэж байна
17. display(x,y);
18. //Хэрэглэгчийн оруулсан хоёр тоог хадгалж буй хувьсагчдийн дамжуулж утгийг нь заалтаар сольж байна
19. swap(x, y);
20. //хоёр хувьсагчийн утга өөрчлөгдсөн эсэхийг хэвлэж шалгаж байна
21. display(x, y);
22. }
24. /\*дамжуулсан хоёр хувьсагчийн утгийг
25. x- дамжуулсан х-г заах заалтан хувьсагч
26. y- дамжуулсан y-г заах заалтан хувьсагч
27. X,y –г заалтан хувьсагч болгосноор санах ой хэмнэж байгаа юм.
28. \*/
29. void swap(int &x, int &y) {
30. //int төрлийн temp хувьсагчид x заалтан хувьсагчийн утгийг хадгалж авч байна.
31. int temp = x;
32. //x-д заалтан хувьсагчид y-д байгаа утгийг оноож байна
33. x = y;
34. //x заалтан хувьсагчид y-д байгаа утга буюу y-д байсан утгийг оноож байна
35. y = temp;
36. }
37. //дамжуулсан хоёр хаягийн утгийг хэвлэх функц зарлаж байна
38. //x,y заалтан хувьсагч үүсгээд x,y –г зааж байна
39. void display(int & x, int & y) {
40. //x,yзаалтан хувьсагчийн утгийг cout объектийн << операторыг ашиглаж хэвлэж байна.
41. cout << "x= " << x << " y=" << y << endl;
42. }