# Классын тухай ойлголт (Лаборатори №3)

Д.Самдан (16B1SEAS2873)

ХШУИС, ПХ-ийн 3-р түвшин, pri.samdan@gmail.com

# **1. ОРШИЛ**

Класс, объектын талаар судалж, тэдгээрийг бодит амьдрал дээр хэрхэн ашиглах талаар ажилчин объектыг С++ хэлийг ашиглан тодорхойлно. Мөн классын гишүүн өгөгдөл, функц, хандалтын түвшин гэж юу болох талаар энэ лабораторийн ажлаар өгүүлнэ.

# **2. ЗОРИЛГО**

Ажилчин нэртэй объектыг 3-р лабораторын 5-р дасгалд өгөгдсөн класс диаграмын дагуу зарлаж, түүний гишүүн функцуудыг нь тодорхойлно. Үүний тулд дараах зорилтуудыг тавьж ажилласан:

* Класс, объектын талаар онолын судалгаа хийх
* Классыг тодорхойлох
  + Гишүүн өгөгдлүүдийг тодорхойлох
  + Гишүүн функцуудыг тодорхойлох
  + Хандалтын түвшинг тодорхойлох
* Гишүүн функцуудыг зарлах
* Объект зарлах
* Объектоор дамжин тодорхойлсон функцуудээ ажиллуулах

# **3. ОНОЛЫН СУДАЛГАА**

**Класс, Объект**

Класс нь хэрэглэгчийн тодорхойлсон өгөгдлийн төрөл юм. Энэ нь гишүүн өгөгдөл болон гишүүн функцийг дотороо агуулдаг. Класс нь обектын загвар юм.[2]

**Классын тодорхойлолт**

class классын нэр

{

private / protected / public: // Хандалтын түвшин

Гишүүн өгөгдөл ( data member )

private / public: // Хандалтын түвшин

Гишүүн функц ( member function )

}

Class гэсэн тусгай түлхүүр үгийн хойно зарлах гэж буй классын нэр, түүний дараа багц коммандын хаалт нээж гишүүн өгөгдөл болон гишүүн функцийг зарлаж өгнө. Гишүүн өгөгдөл, функцийг тодорхойлохоос өмнө хандалтын түвшинг тодорхойлж болно. Private, Protected, Public гэсэг гэравын адь нэгийг нь хэрэглээгүй үед, классын бүх гишүүн private байна.[1]

* 1. Гишүүн өгөгдөл

Тухайн классын шинж тэмдэгүүд байна.

* 1. Гишүүн функц

Гишүүн функц нь классынхаа гишүүн өгөгдлийг боловсруулах, бусад функцтэй өгөгдөл, мэдээлэл солилцох интерфейс юм. Мөн гишүүн функцийг классын тодорхойлолтын дотор болон гадна тодорхойлж болно.

1. **Хандалтын түвшин**

Класс түүний гишүүд рүү хандах хандалтыг private, public, protected гэсэн гурван түвшинд авч үздэг. Эдгээр хандалтын түвшин нь гишүүд хаанаас хандаж болох, болохгүйг заана.[1]

* Private: Гишүүн рүү зөвхөн тухайн класс дотороос хандаж болно.
* Public: Гишүүн рүү тухайн классын гаднаас хандаж болно.
* Protected: Private түвшинтэй адил. Гэхдээ удамших талаар ялгаатай.

1. **Өгөгдлийн далдлал, битүүмжлэл**

Өгөдлийн далдлал нь объектын утга далд байна гэсэг утгатай гэвч энэ нь түүнийг бусдаас хамгаалах нууц гэсэн үг биш. Харин объектын гишүүн өгөгдөл рүү зөвхөн гишүүн функцээр дамужуулан хандахыг хэлнэ. Ингэснээр гишүүн өгөгдлийн санамсаргүй өөрчлөлтөөс хамгаална.

Өгөгдөл болон түүнд үйлдэл хийх функцийг хамтдаа авч үздэг нь OO технологийн чухал шинж юм. Иймд битүүмжлэл гэдэг нь функц, өгөгдлийг тодорхой түвшинд нээлттэй биш объектын хүрээнд хамтад нь холбоотойгоор авч үздэг ойлголт юм. [1]

1. **Объект байгуулах**

Класс тодорхойлсны дараагаар түүний хэдэн ч хувьсагч зарлаж болох ба ийм хувьсагчийг объект, зарлах үйлийг нь объект байгуулах гэнэ.

class\_name object\_name;

Дээрх комманд нь object\_name гэсэн объектд зориулж С++ компайлер class\_name-н гишүүн өгөгдлийн дагуу ой бэлдэнэ.

class\_name ob1, ob2, ob2;

ob1, ob2, ob3 гэсэн гурван объект зэрэг зарлаж болно. Энэ тохиолдолд гишүүн өгөгдөл классын объект бүрд, харин гишүүн функц классын хэмжээнд ганц удаа байгуулагдана. Нэг классын объектууд классын гишүүн функцийг нь дундаа хэрэглэнэ. [1]

1. **Гишүүн өгөгдөл рүү хандах**

Классыг тодорхойлж өгөхөд хандалтын түвшингөөс хамаарч өөр өөрөөр хандана.

Хэрэв гишүүн өгөгдөл public түвшинтэй бол гишүүн өгөгдөл рүү тухайн классыг объектоор дамжуулан шууд сонголтын цэг (.) оператор ашиглан хандаж болно.

object\_name.data\_name

object\_name гэсэн объектд байгаа data\_name гэсэн гишүүн өгөгдөл рүү хандаж байна.

Private, protected үед дээрх шиг хандаж болохгүй юм. Хэрэв тэгэх юм бол системд доорх алдаа гарна.

*“data\_name cannot be accessible”*

Шалтгаан нь тухайн объект private, protected шинжтэй учир классын биш функцийн хувьд нууцлагдамал байдаг. Нууцлагдмал классын гишүүн өгөгдөл рүү тухайн классд орших public функцээр дамжуулан хандаж болно. [1]

1. **Гишүүн функц рүү хандах**

Классын гишүүн функц рүү тухайн классын объектоор нь дамжуулж хандана.

Ингэхдээ объектын нэрийг гишүүн функцийн нэрээс шууд сонголтын цэг (.) оператороор зааглаж бичнэ.

EMPLOYEE классын GETDATA(), SHOWDATA() функцүүдийг emp объектоор дамжуулж дараах байдлаар дууддаг. [1]

emp.getdata();

emp.showdata();

1. **Гишүүн функцийг тодорхойлох**

Гишүүн функцийг тухайн классын дотор эсвэл классын гадна тодорхойлж болно. Хэрэв классын гадна тодорхойлсон тохиолдолд доор үзүүлсэн загварын дагуу тодорхойлно.

            return\_type class\_name :: function\_name ( parameters ) { // code }

 Жирийн функцийн тодорхойлолттой төстэй юм. Үүндээ class\_name-г ( :: ) буюу үйлчлэх хүрээний оператороор залгаж бичнэ. Залгаж өгссөнөөрөө тухайн функц class\_name-д харьяалагдаж, түүний объектод үйлчилнэ гэдэг утгыг илэрхийлнэ.

**4. Хэрэгжүүлэлт**

Хавсралт employee.h-д классын тодорхойлолт байна. 4-12 мөрт гишүүг өгөгдлүүдээ тодорхойлсон. Түүний дараа 13-25 мөрт гишүүн функцүүдийг нь зарлаж өгсөн.

Хавсралт employee.cpp-д employee.h-н зарлаж өгсөн функцүүдийг нь тодорхойлж өгнө. Функцийг нь тодорхойлохоос өмнө employee.h файлыг #include хийж классын тодорхойлолтыг авна. Функцийг тодорхойлохоос өмнө тухайн функц employee.h-д зарласан employee классынх гэж зааж өгөх :: үйлчлэх хүрээний оператор залгана. Үүний дараа функцүүдээ тодорхойлж бичнэ.

employee.cpp-г С++ компайлер ашиглан объект файл үүсгэн. Ингэснээр employee.cpp-д бичсэн функцүүдийг тодорхойлолтыг хэрэглэгчээс нууж чадна. Хэрэглэгч зөвхөн employee.h болон үүссэн объект файлтай байхад үүсгэсэн employee классыг ашиглаж болно.

Объект файл үүсгэх:

            g++ -c employee.cpp

Үр дүн:

            employee.o

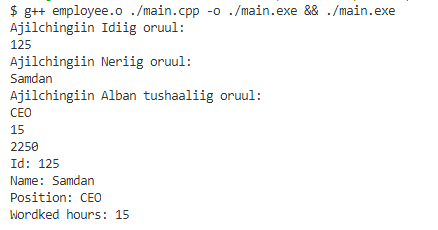
Үүсгэсэн employee классыг ашиглах нь:

Хавсралт main.cpp-д хэрэглэгчийн main функцийн дээр employee.h файлыг #include хийж классын тодорхойлолтыг авна. main функц дотор employee классыг ашиглан. Эхлээд 6-р мөрт emp гэсэн employee төрлийн объект байгуулна. Тухай объектоор хандан employee.h-д зарласан функцүүдийг дуудаж ажиллуулна (9, 11, 18, 21, 24 мөрүүдэд).

Хэрэглэгч main функцээ бичээд дууссаны дараа .exe файл үүсгэхдээ өгсөн объект файлтай хамт С++ компайлер-г ашиглан компайл хийнэ.

            g++ employee.o ./main.cpp -o ./main.exe

Үүсэн main.exe-г ажиглуулна.



# **5. ДҮГНЭЛТ**

# Класс ба объектын талаар судалж үзсэний үр дүнд гишүүн өгөгдөл, гишүүн функц, тэдгээрт хэрхэн хандах, хандалтын түвшин, өгөгдөл битүүмжлэх зэрэг ойлголтуудыг дэлгэрэнгүй мэлэж авлаа. Класс ба объектыг хэрэглэснээр амьдрал дээрх зүйлсийг програмчлахад илүү хялбар болж байна. Учир нь класс нь бодит ертөнц дээрх юмс үзэгдэлийг програмчлахад зөвхөн шинж чанар үйл хөдлөлийг нь судалж улмаар түүнд тохирсон код функц бичих боломжтой болж байгаа юм. Мэдээлэл технологийн эрин зуунд амьдарч байгаа бид хэрэглэгчид хүртээмжтэй ашиглахад хялбар программ хангамж бүтээх нь бидний зорилго юм.Үүний тулд класс обьект ашиглах нь шилдэг арга болох болно.

# **6. АШИГЛАСАН МАТЕРИАЛ**

1. Объект хандлагат технологийн С++ програмчлал, Ж.Пүрэв, 2008, Улаанбаатар.

2. C++ Classes and Objects, <https://www.geeksforgeeks.org/c-classes-and-objects/>

# **7. ХАВСРАЛТ**

**employee.h**

1. #include <iostream>
2. using namespace std;
3. class employee {
4. public:
5. // Гишүүн ажилчны дугаар
6. int id;
7. // Гишүүн ажилчны нэр
8. char name[20];
9. // Гишүүн ажилчны албан тушаал
10. char position[20];
11. // Гишүүн ажилчны ажилсан цаг
12. float worked\_hour;
13. public:
14. // Гишүүн ажилчинд анхны утга оноож үүсгэнэ.
15. void init();
16. // Гишүүн ажилчны мэдээлэл дэлгэцэнд хэвлэх
17. void print();
18. // Гишүүн ажилчны мэдээлэл
19. void read();
20. // Гишүүн ажилчны цалинг ажилласан цагаар нь бодож буцаана.
21. float calculate\_salary();
22. // Захирлын цалинг бодох
23. float calculate\_salary\_ceo();
24. // Гишүүн ажилчны ажилласан цагийг нэмнэ
25. bool add\_worked\_hour(float hour);
26. };

**employee.cpp**

#include "./employee.h"

#include <string.h>

/\*

Гишүүн ажилчинд анхны утга оноож өгнө

Params:

void

Return:

void

\*/

void employee::init()

{

id = 0;

strcpy(name, "");

strcpy(position, "employee");

worked\_hour = 0.0;

}

/\*

Гишүүн ажилчны мэдээлэлийг дэлгэцэнд хэвлэнэ

Params:

void

Return:

void

\*/

void employee::print()

{

cout << "Id: " << id << endl;

cout << "Name: " << name << endl;

cout << "Position: " << position << endl;

cout << "Wordked hours: " << worked\_hour << endl;

}

/\*

Гишүүн ажилчны мэдээлэл

Params:

void

Return:

void

\*/

void employee::read()

{

cout << "Ajilchingiin Idiig oruul: " << endl;

cin >> id;

cout << "Ajilchingiin Neriig oruul: " << endl;

cin >> name;

cout << "Ajilchingiin Alban tushaaliig oruul: " << endl;

cin >> position;

}

/\*

Гишүүн ажилчны цалинг ажилласан цагаар нь бодож буцаана.

Params:

void

Return:

void

\*/

float employee::calculate\_salary()

{

float salary = worked\_hour \* 100;

if (strcmp(position, "CEO") == 0)

return salary + calculate\_salary\_ceo();

return salary;

}

/\*

Захирлын цалинг бодох

Params:

void

Return:

void

\*/

float employee::calculate\_salary\_ceo()

{

return worked\_hour \* 50;

}

/\*

Гишүүн ажилчны ажилласан цагийг нэмнэ

Params:

float hour - Ажилчингын ажилсн цаг

Return:

bool - Зөв утга уруулсан үед 1 бусад үед 0

\*/

bool employee::add\_worked\_hour(float hour)

{

if (hour > 0 && hour < 24)

{

worked\_hour += hour;

return true;

}

return false;

}

**main.cpp**

#include "./employee.h"

int main()

{

// Ажилчин төрлйин emp гэсэн объект байгуулна

employee emp;

// Ажилчин объект emp-р дамжуулан init функцийг дуудна

emp.init();

// Ажилчин объект emp-р дамжуулан read функцийг дуудна

emp.read();

// Гишүүн ажилчны ажилсан цагийг хадаглах бодит тоон хувьсагч зарлана

float hour;

// Гишүүн ажилчны ажилсан цагийг гараас оруулна

cin >> hour;

// Гишүүн ажилчны ажилсан цагийг add\_worked\_hour-г ашиглна нэмнэ emp объектын ажилсан цаг дээр нэмнэ

emp.add\_worked\_hour(hour);

// Ажилчин объект emp-р дамжуулан calculate\_salary функцийг ашиглан хэвлэнэ

cout << emp.calculate\_salary() << endl;

// Ажилчин объект emp-р дамжуулан print функцийг ашиглан emp объектын мэдээллийг хэвлэнэ

emp.print();

}