# **Удамшлын харьцаа болон бүрдэл харьцааны онолын талаар судлаж объект хандлагат програмчлалд хэрхэн хэрэгжүүлэх талаар судлах**

# (Лаборатори №11)

Г. Саруул (20B1NUM2095)

ХШУИС, МКУТ-ийн Компьютерийн ухааны  хөтөлбөрийн 2-р түвшиний оюутан

# **1. ОРШИЛ/УДИРТГАЛ**

Удамшлын үүсгэх харьцаа болон бүрдэл харьцааны талаар дэлэнгрэнгүй судлаж объект хандлагат програмчлалд хэрхэн хэрэгжүүлж болох талаар судалсан. Мөн бүрдэл харьцаан дотроо хориглолт буюу constraint гэж юу болох талаар судалсан. Объект хандлагат хэл болох С++ хэлэн дээр удамшлын үүсгэх харьцаа болон бүрдэл харьцааны кодыг бичиж хэрэгжүүлсэн.

# **2. ЗОРИЛГО**

Удамшлын үүсгэх харьцаа болон бүрдэл харьцааны талаар судлан объект хандалагат програмчлалын хэл дээр бичиж хэрэгжүүлнэ. Үүний тулд дараах зорилтуудыг тавьж ажилласан:

1. Удамшил ямар харьцаа үүсгэх талаар судлах,
2. Бүрдэл харьцаа гэж юу болох талаар судлах,
3. Бүрдэл харьцааны хориглолт (constraint) гэж юу болох талаар судлах,
4. Судалсан харьцаануудаа объект хандлагат програмчлалын хэлэнд бичиж хэрэгжүүлэх.

# **3. ОНОЛЫН СУДАЛГАА**

## **3.1 Удамшил ямар харьцаа үүсгэдэг вэ? Объект хандлагат программчлалд хэрхэн хэрэгжүүлдэг вэ?**

Удамшил гэдэг нь нэг классаас өөр нэг класс үүсэхийг хэлнэ. Удамшсан класс буюу дэд класс нь эх классынхаа гишүүн өгөгдөл болон гишүүн функцыг бүгдийн өвлөж авдаг. Мөн өөрийн гэсэн нэмэлт гишүүн өгөгдөл болон гишүүн функцуудтай байж болно. Тэгвэл тэдгээр классын хоёуланд нь байх зарим нэгэн ерөнхий шинж байж болно. Эх ба дэд класс хооронд байх энэ харьцаа бол “is a kind of” буюу (...-ны) төрлийнх юм. Ерөнхийллөөс нарийсгал холбоо үүсгэж байвал᠌ “is a” харьцаа болно.

## **3.2 Бүрдэл харьцаа гэж юу вэ? Объект хандлагат программчлалд хэрхэн хэрэгжүүлдэг вэ?**

Бүрдэл харьцааг товчоор хэлвэл “has a” (-тай) харьцаа юм. Ийм харьцаа нь нэг классын объект өөр классын объектыг дотроо агуулж байвал, өөрөөр хэлбэл нэг классын объект өөр классын гишүүн болж байвал үүснэ. Хамгийн ойр жишээ бол машин юм. Машин нь хөдөлгүүртэй болон дугуйнуудтай гэж үзнэ. Тэгвэл машин гэдэг класс нь дугуй класс болон хөдөлгүүр классын объектуудыг агуулна гэж ойлгож болно.

class car : public vehicle[1] {

Engine myEngine;

};

Харин эсэргээрээ дугуйн класс болон хөдөлгүүрийн классаас машины класс удамшуулж болохгүй.

## **3.3 Бүрдэл харьцааны хориглолт (constrain) гэж юу вэ?**

Бүрдэл харьцаан хориглолт гэдэг нь классууд өөр бусад классын объектыг агуулахад хориглолт хийж байгааг хэлнэ. Жишээ нь:

1. 0…1 (Хамгийн багадаа 0 буюу байхгүй, хамгийн ихдээ 1 ширхэгийг агуулна)
2. 0…n (Хамгийн багадаа 0 буюу байхгүй, хамгийн ихдээ хэд ч байж болно)
3. 1…1 (Хамгийн багадаа 1, хамгийн их дээ 1 буюу ямар ч үед 1ийг агуулна)
4. 1…n (Хамгийн багадаа 1, хамгийн ихдээ хэд ч байж болно)

Дээрх бүрдэл харьцааны хориглолт нь ямар нэгэн классд хамгийн багадаа болон хамгийн ихдээ хэдэн өөр классын объект агуулагдаж болохы заана гэсэн үг.

# **4. ХЭРЭГЖҮҮЛЭЛТ**

1. Person нэртэй классыг үүсгэж гишүүн өгөгдөл болон гишүү фунцкуудийг тус тус бичсэн.

Гишүүн өгөгдөл: Name (char), SSNum (char), Age (int)

Гишүүн функц: char getName(), char SSNum(), int getAge(), void setName(), void setSSNum(), void setAge(), Байгуулагч функц, Устагч функц, void showPersonData() гэсэн функцуудыг тус тус бичсэн.

**Бичсэн код:**

class Person{

private:

char \*name;

char \*ssnum;

int age;

public:

Person(){

name = new char(8);

strcpy(name, "default name");

ssnum = new char(8);

strcpy(ssnum, "default ssnum");

age = 0;

// cout << "Person class's anhdagch function created" << endl;

}

Person(char \*ner, char \*registerDugaar, int nas){

name = new char(strlen(ner) + 1);

strcpy(name, ner);

ssnum = new char(strlen(registerDugaar) + 1);

strcpy(ssnum, registerDugaar);

age = nas;

// cout << "Person class's constructer function created" << endl;

}

char \*getName(){

return name;

}

char \*getSSNum(){

return ssnum;

}

int getAge(){

return age;

}

void setName(char \*ner){

delete name;

name = new char(strlen(ner) + 1);

strcpy(name, ner);

}

void setSSNum(char \*registerDugaar){

delete ssnum;

ssnum = new char(strlen(registerDugaar) + 1);

strcpy(ssnum, registerDugaar);

}

void setAge(int a){

age = a;

}

void showPersonData(){

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

cout << "Register:\t" << ssnum << endl;

}

~Person(){

delete name;

delete ssnum;

}

};

2. Spouse нэртэй классыг Person нэртэй классаас удамшуулж бичсэн. Гишүүн өгөгдөл нь AnniversaryDate. Гишүүн функцууд нь Анхдагч байгуулагч, Параметртэй байгуулагч, void showSpouseData(), void setAnniversaryDate(int a), int getAnniversaryDate() гэсэн фунцкуудыг тус тус бичсэн.

**Бичсэн код:**

class Spouse: public Person{

private:

int AnniversaryDate;

public:

Spouse(): Person(){

AnniversaryDate = 0;

}

Spouse(int date, char \*ner, char \*ssnum, int nas): Person(ner, ssnum, nas){

AnniversaryDate = date;

cout << "Spouse class's constructe created" << endl;

}

int getAnniversaryDate(){

return AnniversaryDate;

}

void setAnniversaryDate(int a){

AnniversaryDate = a;

}

void showSpouseData(){

showPersonData();

cout << "Gerlesen jil:\t" << AnniversaryDate << endl;

}

};

3. Child нэртэй классыг Person нэртэй классаас удамшуулж үүсгэсэн. Гишүүн өгөгдөл нь char \*favoriteToy. Гишүүн функцууд нь Анхдагч байгуулагч, Параметртэй байгуулагч, char getFavoriteToy(), void setFavoriteToy(char \*togloom), void showChildData(), устгагч функцуудыг тус тус бичсэн.

**Бичсэн код:**

class Child: public Person{

protected:

char \*favoriteToy;

public:

Child(): Person("Defualt name", "Defualt ssnum", 0){

favoriteToy = new char(8);

strcpy(favoriteToy, "defualt");

}

Child(char \*togloom, char \*ner, char \*ssnum, int nas): Person(ner, ssnum, nas){

favoriteToy = new char(strlen(togloom) + 1);

strcpy(favoriteToy, togloom);

cout << "Child class's constructe created" << endl;

}

char getFavoriteToy(){

return \*favoriteToy;

}

void setFavoriteToy(char \*togloom){

delete favoriteToy;

favoriteToy = new char(strlen(togloom)+1);

strcpy(favoriteToy, togloom);

}

void showChildData(){

showPersonData();

cout << "Durtai togloom:\t" << favoriteToy << endl;

}

~Child(){

delete favoriteToy;

}

};

4. Employee нэртэй классыг Person классаас удамшуулж үүсгэсэн. Гишүүн өгөгдөлүүд нь char \*companyID, char \*title, int startDate, Spouse ehner, JobDescription ajil, Division bairlal, vector<Child> child гэсэн гишүүн өгөгдөл болон бүрдэл харьцаануудыг тус тус бичсэн. Гишүүн функцууд нь Анхдагч байгуулагч, Параметртэй байгуулагч, setter, getter функцууд, void showData(), устгагч байгуулагч гэх мэт тус тус бичсэн.

**Бичсэн код:**

class Employee: public Person{

protected:

char \*companyID;

char \*title;

int startDate;

Spouse ehner;

vector<JobDescription> ajil;

Division bairlal;

vector<Child> child;

public:

Employee(): Person(){

companyID = new char(8);

strcpy(companyID, "defualt company ID");

title = new char(8);

strcpy(title, "defualt title");

startDate = 0;

}

Employee(char \*kompaniID, char \*Kompani, int ehelsenOgnoo, char \*ner, char \*ssnum, int nas, Division division, JobDescription job): Person(ner, ssnum, nas){

companyID = new char(strlen(kompaniID) + 1);

strcpy(companyID, kompaniID);

title = new char(strlen(Kompani) + 1);

strcpy(title, Kompani);

addJobDescription(job);

// child = new vector<Child>();

startDate = ehelsenOgnoo;

setDivision(division);

cout << "Employee class parametr constructer created" << endl;

}

char getCompanyID(){

return \*companyID;

}

char getTitle(){

return \*title;

}

int getStartDate(){

return startDate;

}

Spouse getSpouce(){

return ehner;

}

vector<Child> getChild(){

return child;

}

vector<JobDescription> getJob(){

return ajil;

}

Division getDivision(){

return bairlal;

}

void setCompanyID(char \*id){

delete companyID;

companyID = new char(strlen(id)+1);

strcpy(companyID, id);

}

void setTitle(char \*garchig){

delete title;

title = new char(strlen(garchig)+1);

strcpy(title, garchig);

}

void setStartDate(int a){

startDate = a;

}

void setSpouse(Spouse spouse){

if(strcmp(getName(), "default name") != 0){

ehner = spouse;

}

}

void addChild(Child child){

this->child.push\_back(child);

}

void addJobDescription(JobDescription job){

this->ajil.push\_back(job);

}

void setDivision(Division division){

bairlal = division;

}

void showData(){

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Huviin-medeelel\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

// showPersonData();

cout << "Name:\t\t" << getName() << endl;

cout << "Register:\t" << getSSNum() << endl;

cout << "Nas:\t\t" << getAge() << endl;

cout << "-----------------------Ajil-alba-------------------" << endl;

cout << "Company ID:\t" << companyID << endl;

cout << "Title:\t\t" << title << endl;

cout << "Start date:\t" << startDate << endl;

cout << "Division\t" << bairlal.getDivisionName() << endl;

for(int i=0; i<ajil.size(); i++){

ajil[i].showJob();

}

// cout << ajil.size() << endl;

// if(ehner.getAnniversaryDate() != NULL)

cout << "-----------------------Gergii-------------------" << endl;

ehner.showSpouseData();

cout << "-----------------------Huuhed-------------------" << endl;

for(int i=0; i<child.size(); i++){

child[i].showChildData();

}

// cout << child.size() << endl;

}

~Employee(){

delete companyID;

delete title;

cout << "Employee destroyed" << endl;

}

};

5. Division болон JobDescription гэсэн нэртэй классуудыг тус тус үүсгэсэн. Гишүүн өгөгдөлүүдийг хаяган оператороор тодорхойлж өгсөн. Мөн тус бүрт нь анхдагч байгуулагч болон параметртэй байгуулагч, setter, getter функцуудыг тус тус бичсэн.

**Бичсэн код:**

class Division{

private:

char \*divisionName;

public:

Division(){

divisionName = new char(8);

strcpy(divisionName, "defualt");

}

Division(char \*bairlal){

divisionName = new char(strlen(bairlal)+1);

strcpy(divisionName, bairlal);

}

char \*getDivisionName(){

return divisionName;

}

void setDivisionName(char \*divisionNer){

delete divisionName;

divisionName = new char(strlen(divisionNer)+1);

strcpy(divisionName, divisionNer);

}

~Division(){

delete divisionName;

}

};

class JobDescription{

private:

char \*description;

public:

JobDescription(){

description = new char(8);

strcpy(description, "defualt");

}

JobDescription(char \*jobTailbar){

description = new char(strlen(jobTailbar) + 1);

strcpy(description, jobTailbar);

}

char \*getDescription(){

return description;

}

void setDescription(char \*desc){

delete description;

description = new char(strlen(desc) + 1);

strcpy(description, desc);

}

void showJob(){

cout << "Job:\t\t" << description << endl;

}

~JobDescription(){

delete description;

}

};

Main функц дотроо класс болгон дээр хэд хэдэн объект үүсгэж Employee классын объектуудад Child, Spouse, JobDescription, Division классуудын объектуудыг тохируулж өгсөн. Эцэст нь Employee классын объектуудын бүх мэдээллийг хэвлэж харуулсан.

# **5. ДҮГНЭЛТ**

Удамшлын үүсгэх харьцаа болон бүрдэл харьцаан талаар дэлэнгрэнгүй мэдэж авсанаар та классыг классаас удамшуулж, өөр классын объектыг ямар нэг классын гишүүн өгөгдөл болгон ашиглаж чадахаас гадна маш их цаг хугацаа болон өртөг зэргийг хэмнэж чадах юм. Учир нь удамшил болон бүрдэл харьцаа зэргийг ашигласнаар кодны бичиглэл багасхаас гадна объект хандлагад програмчлалын талаар өргөн мэдлэгтэй болох юм.

# **6. АШИГЛАСАН МАТЕРИАЛ**

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

[2]. [Composition over inheritance - Wikipedia](https://en.wikipedia.org/wiki/Composition_over_inheritance)

[3]. [Difference Between Composition and Inheritance | Difference Between](http://www.differencebetween.net/technology/difference-between-composition-and-inheritance/)

[4]. [Favoring Composition Over Inheritance In Java With Examples - GeeksforGeeks](https://www.geeksforgeeks.org/favoring-composition-over-inheritance-in-java-with-examples/)

[5]. [vector - C++ Reference (cplusplus.com)](http://cplusplus.com/reference/vector/vector/)

[6]. [C++ Library - <vector> (tutorialspoint.com)](https://www.tutorialspoint.com/cpp_standard_library/vector.htm)

[7]. [C++ Vector Size or Length (tutorialkart.com)](https://www.tutorialkart.com/cpp/cpp-vector-size/#:~:text=To%20get%20the%20size%20of%20a%20C%2B%2B%20Vector%2C,then%20find%20its%20length%20using%20size%20%28%29%20function.)

# **7. ХАВСРАЛТ**

#include<iostream>

#include<string.h>

#include <vector>

using namespace std;

class Person{

private:

char \*name;

char \*ssnum;

int age;

public:

Person(){

name = new char(8);

strcpy(name, "default name");

ssnum = new char(8);

strcpy(ssnum, "default ssnum");

age = 0;

// cout << "Person class's anhdagch function created" << endl;

}

Person(char \*ner, char \*registerDugaar, int nas){

name = new char(strlen(ner) + 1);

strcpy(name, ner);

ssnum = new char(strlen(registerDugaar) + 1);

strcpy(ssnum, registerDugaar);

age = nas;

// cout << "Person class's constructer function created" << endl;

}

char \*getName(){

return name;

}

char \*getSSNum(){

return ssnum;

}

int getAge(){

return age;

}

void setName(char \*ner){

delete name;

name = new char(strlen(ner) + 1);

strcpy(name, ner);

}

void setSSNum(char \*registerDugaar){

delete ssnum;

ssnum = new char(strlen(registerDugaar) + 1);

strcpy(ssnum, registerDugaar);

}

void setAge(int a){

age = a;

}

void showPersonData(){

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

cout << "Register:\t" << ssnum << endl;

}

~Person(){

delete name;

delete ssnum;

}

};

class Spouse: public Person{

private:

int AnniversaryDate;

public:

Spouse(): Person(){

AnniversaryDate = 0;

}

Spouse(int date, char \*ner, char \*ssnum, int nas): Person(ner, ssnum, nas){

AnniversaryDate = date;

cout << "Spouse class's constructe created" << endl;

}

int getAnniversaryDate(){

return AnniversaryDate;

}

void setAnniversaryDate(int a){

AnniversaryDate = a;

}

void showSpouseData(){

showPersonData();

cout << "Gerlesen jil:\t" << AnniversaryDate << endl;

}

};

class Child: public Person{

protected:

char \*favoriteToy;

public:

Child(): Person("Defualt name", "Defualt ssnum", 0){

favoriteToy = new char(8);

strcpy(favoriteToy, "defualt");

}

Child(char \*togloom, char \*ner, char \*ssnum, int nas): Person(ner, ssnum, nas){

favoriteToy = new char(strlen(togloom) + 1);

strcpy(favoriteToy, togloom);

cout << "Child class's constructe created" << endl;

}

char getFavoriteToy(){

return \*favoriteToy;

}

void setFavoriteToy(char \*togloom){

delete favoriteToy;

favoriteToy = new char(strlen(togloom)+1);

strcpy(favoriteToy, togloom);

}

void showChildData(){

showPersonData();

cout << "Durtai togloom:\t" << favoriteToy << endl;

}

~Child(){

delete favoriteToy;

}

};

class Division{

private:

char \*divisionName;

public:

Division(){

divisionName = new char(8);

strcpy(divisionName, "defualt");

}

Division(char \*bairlal){

divisionName = new char(strlen(bairlal)+1);

strcpy(divisionName, bairlal);

}

char \*getDivisionName(){

return divisionName;

}

void setDivisionName(char \*divisionNer){

delete divisionName;

divisionName = new char(strlen(divisionNer)+1);

strcpy(divisionName, divisionNer);

}

~Division(){

delete divisionName;

}

};

class JobDescription{

private:

char \*description;

public:

JobDescription(){

description = new char(8);

strcpy(description, "defualt");

}

JobDescription(char \*jobTailbar){

description = new char(strlen(jobTailbar) + 1);

strcpy(description, jobTailbar);

}

char \*getDescription(){

return description;

}

void setDescription(char \*desc){

delete description;

description = new char(strlen(desc) + 1);

strcpy(description, desc);

}

void showJob(){

cout << "Job:\t\t" << description << endl;

}

~JobDescription(){

delete description;

}

};

class Employee: public Person{

protected:

char \*companyID;

char \*title;

int startDate;

Spouse ehner;

vector<JobDescription> ajil;

Division bairlal;

vector<Child> child;

public:

Employee(): Person(){

companyID = new char(8);

strcpy(companyID, "defualt company ID");

title = new char(8);

strcpy(title, "defualt title");

startDate = 0;

}

Employee(char \*kompaniID, char \*Kompani, int ehelsenOgnoo, char \*ner, char \*ssnum, int nas, Division division, JobDescription job): Person(ner, ssnum, nas){

companyID = new char(strlen(kompaniID) + 1);

strcpy(companyID, kompaniID);

title = new char(strlen(Kompani) + 1);

strcpy(title, Kompani);

addJobDescription(job);

// child = new vector<Child>();

startDate = ehelsenOgnoo;

setDivision(division);

cout << "Employee class parametr constructer created" << endl;

}

char getCompanyID(){

return \*companyID;

}

char getTitle(){

return \*title;

}

int getStartDate(){

return startDate;

}

Spouse getSpouce(){

return ehner;

}

vector<Child> getChild(){

return child;

}

vector<JobDescription> getJob(){

return ajil;

}

Division getDivision(){

return bairlal;

}

void setCompanyID(char \*id){

delete companyID;

companyID = new char(strlen(id)+1);

strcpy(companyID, id);

}

void setTitle(char \*garchig){

delete title;

title = new char(strlen(garchig)+1);

strcpy(title, garchig);

}

void setStartDate(int a){

startDate = a;

}

void setSpouse(Spouse spouse){

if(strcmp(getName(), "default name") != 0){

ehner = spouse;

}

}

void addChild(Child child){

this->child.push\_back(child);

}

void addJobDescription(JobDescription job){

this->ajil.push\_back(job);

}

void setDivision(Division division){

bairlal = division;

}

void showData(){

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Huviin-medeelel\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

// showPersonData();

cout << "Name:\t\t" << getName() << endl;

cout << "Register:\t" << getSSNum() << endl;

cout << "Nas:\t\t" << getAge() << endl;

cout << "-----------------------Ajil-alba-------------------" << endl;

cout << "Company ID:\t" << companyID << endl;

cout << "Title:\t\t" << title << endl;

cout << "Start date:\t" << startDate << endl;

cout << "Division\t" << bairlal.getDivisionName() << endl;

for(int i=0; i<ajil.size(); i++){

ajil[i].showJob();

}

// cout << ajil.size() << endl;

// if(ehner.getAnniversaryDate() != NULL)

cout << "-----------------------Gergii-------------------" << endl;

ehner.showSpouseData();

cout << "-----------------------Huuhed-------------------" << endl;

for(int i=0; i<child.size(); i++){

child[i].showChildData();

}

// cout << child.size() << endl;

}

~Employee(){

delete companyID;

delete title;

cout << "Employee destroyed" << endl;

}

};

int main(){

Division div1("Staff"), div2("Ceo");

JobDescription job1("Driver"), job2("Teacher"), job3("Chef");

Child huuhed1("CSGO", "Anujin", "PD2020\*\*\*\*", 3);

Child huuhed2("Basketball", "Tenuun", "PD2019\*\*\*\*", 4);

Child huuhed3("Dota2", "Burte", "PD2018\*\*\*\*", 5);

Spouse ehner1(2019, "Ujin", "RT0223\*\*\*\*", 19);

Spouse ehner2(2017, "Bolormaa", "WP0229\*\*\*\*", 19);

// Employee Bilguun, Bold;

Employee Bilguun("111", "Steam", 2020, "Bilguun", "PD0225\*\*\*\*", 19, div1, job1);

Employee Bold("222", "Google", 2021, "Bold", "RI0235\*\*\*\*", 18, div2, job2);

Bilguun.setSpouse(ehner1);

Bold.setSpouse(ehner2);

Bilguun.addChild(huuhed1);

Bilguun.addChild(huuhed2);

Bold.addChild(huuhed3);

Bilguun.addJobDescription(job3);

Bilguun.showData();

cout << endl;

Bold.showData();

return 0;

}