C++ хэлний хуулагч байгуулагч, хуулагч функц болон объектын хаяган хувьсагч

Лабораторн 5

Э.Түвшин-Эрдэнэ(23B1NUM0869)
МТЭС, МКУТ-ын “Программ хангамж” хөтөлбөрийн оюутан, tuvshin.erdene25@gmail.com
2025 он

1.Оршил

Энэ лабораторийн ажлаар байгуулагч, устгагч функц хэзээ дуудагддаг болон хуулагч байгуулагч, хуулагч функц, объектын хаяган хувьсагчийн талаар судлав. Жишээ код болон өөрчилсөн ажилчны классыг “хэрэгжүүлэлт” болон “хавсралт” хэсгүүдэд оруулав.

2.Зорилго

Лабораторн 4 -өөр судалсан байгуулагч, устгагч функцуудыг үргэлжлүүлэн судалж, тэдгээр функцууд хэзээ дуудагддаг, хуулагч функц болон хуулагч байгуулагч болон объектын хаяган хувьсагчийн талаар судлан, жишээ кодоор тайлбарлах. Өмнөх лабораториудаар хийсэн ажилчны классыг сайжруулах.

3. Онолын судалгаа

3.1)Байгуулагч болон устгагч функц дуудагдах

Систем объект байгуулахдаа харгалзах классын байгуулагч функцийг автоматаар дуудна.[1]

Объектын үйлчлэх хүрээнээс программын удирдлага гарах үед устгагч функц дуудагдах ба байгуулсан объектын тоогоор устгагч функц дуудагдана.

Устгагч функц тухайн үйлчлэх хүрээнээс гарах эсвэл “delete” үйлдлийг хийхэд автоматаар дуудагдах ба тусгайлан дуудаж өгснөөр алдаа гарах магадлалтай.[2]

3.2)Хуулагч байгуулагч

Хуулагч байгуулагч бол шинэ объект нь түүний өмнөх аль нэг объектын сүүлийн утгыг хуулбарлаж авах боломжийг олгох функц юм. Ийм маягаар байгуулагдах шинэ объект ба утга нь хуулбарлагдах объект хоорондоо зөвхөн нэрээрээ ялгаатай. Ийм байгуулагч нэг параметр авах ба тэр нь класс нэгтэй хуулагдах объектын заалт байна.

Жишээ:

```
//copy constructor
#include<iostream>
//  #include<conio.h>
#include<iomanip>
#include<string.h>
using namespace std;

class Employee{
    private:
        char name[20];
        int basicpay;
        int allowance;
    public:
        Employee(char n[] = "George", int b = 9000, int a = 2000){
            strcpy(name,n);
            basicpay = b;
            allowance = a;
        }
        void showdata();
        void changebasic();
};

void Employee::changebasic(){
    basicpay = 10000;
}
void Employee::showdata(){
    cout<<endl<<setw(20)<<name;
    cout<<setw(8)<<basicpay;
    cout<<setw(12)<<allowance<<endl;
}

void heading(){
    cout<<endl;
    cout<<setw(20)<<"employee name: ";
    cout<<setw(8)<<"basic";
    cout<<setw(12)<<"allowance";
}

int main(){
    Employee emp1;
```

```
    heading();
    emp1.showdata();
    emp1.changebasic();
    emp1.showdata();

    Employee emp2 (emp1);
    emp2.showdata();

    return 0;
}
```

```
employee name:    basic   allowance
         George    9000        2000

         George   10000        2000

         George   10000        2000
```

Хуулагч байгуулагч нь аргумент болох объект руу заалтаар нь хандаж түүний хуулбар объектыг зөвхөн нэг удаа үүсгэнэ.

Хуулагч объектыг функц руу утгаар нь дамжуулахын тулд ашигладаг ба компайлер автоматаар хуулагч объектыг дуудна.

Утга оноох үйлдлээс ялгарах ялгаа нь хуулагч байгуулагч дуудагдах болгондоо шинээр санах ой хуваарилах бол утга оноох үйлдэл шинээр санах ой хуваарилахгүй.[3]

- Оршин байгаа объектоос өөр объект үүсгэхийн тулд хуулагч байгуулагчийг дуудна.
- Утга оноох үйлдлээр аль хэдийн байгуулагч дуудагдан, үүссэн объектод өөр объектын утга онооно.

Шинээр хуулбар объект үүсгэхэд гарааны утга оноох үйлдлийг хялбарчлах ач холбогдолтой.

3.3)Хуулагч функц

Хуулагч функц ашигласнаар хуулах үйлдлийг тохиромжтой байдлаар тохируулах боломжтой. Мөн шинээр хуулсан объектод санах ой бэлддэг “deep copy” хийх боломжтой болдог.

Хэрэв хуулагч функц ашиглахгүйгээр шууд утга оноох үйлдэл ашиглан хуулалт хийвэл тэнцүүгийн тэмдгийн зүүн талын өмнө нь зааж байсан санах ойн хаяг алдагдан memory leak үүсэх эрсдэлтэй.

3.4)Объектын хаяган хувьсагч

Объектын хаяган хувьсагчийг анхдагч өгөгдлийн төрлөн хаяган хувьсагчийн адил зарлана:
<class name> *<pointer name> = &<object name>

Мөн new түлхүүр үгийг ашиглаж болно:
<class name> *<pointer name> = new <class name>// энд гарааны утга оноож болно.
New операторoор үүсгэсэн объектыг “delete” оператор ашиглан санах ойг нь чөлөөлж болно.
Delete <pointer name>

Объектын хаяган хувьсагчаар дамжуулан гишүүн өгөгдөл болон функцэд хандах
Объектон хаяган хувьсагчаас классын гишүүн өгөгдөл эсвэл функц руу 2 аргаар ханддаг:

1. (*<pointer name>).<function||attribute>
2. <pointer name> -> <function||attribute>

3.5)Объектын хаяган хувьсагчийн хүснэгт

Эхлээд объеътын хаяган хувьсагчаас тогтох хүснэгтийг дараах байдлаар үүсгэв:

```
#include <iostream>
#include <string.h>
using namespace std;

class Ant
{
private:
    char name[20];
    float length;
    float width;

public:
    Ant(char n[] = "Ant Bataa", float length = 10, float width=1)
    {
        strcpy(name, n);
        this->length = length;
        this->width = width;
    }
    ~Ant()
    {
        cout << "Ant has retired." << endl;
    }
```

```
    void display(){
        cout<<"name: "<<name<<" length: "<<length<<" width:
"<<width<<endl;
    }
};

int main()
{
    typedef Ant *AntPtr;
    int num_ants;
    cout<<"Enter number of ants: ";cin>>num_ants;
    AntPtr *ants = new AntPtr[num_ants];
    for (int i = 0; i < num_ants; ++i)
    {
        ants[i] = new Ant();
    }

    for (int i = 0; i < num_ants; ++i)
    {
        ants[i]->display();
    }

    return 0;
}
```

```
Enter number of ants: 10
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
name: Ant Bataa length: 10 width: 1
```

Мөн объект дотор хүснэгт бус хаяган хувьсагч ашиглаж үзэв:

```
#include<iostream>
#include<string.h>
using namespace std;

class Employee{
```

```
    private:
        char *name;
        int basicpay;
        int allowance;
    public:
        Employee();
        ~Employee();
        Employee(char *n, int b, int a);
        friend ostream & operator <<(ostream &os, Employee &e);
};

Employee::Employee(){
    name = new char[20];
    strcpy(name,"");
    basicpay = 0;
    allowance = 0;
}
Employee::Employee(char *n, int b, int a){
    name = new char[strlen(n)+1];
    strcpy(name,n);
    basicpay = b;
    allowance = a;
}
Employee::~Employee(){
    cout<<"\n bye bye "<<name<<endl;
    delete name;
}

ostream & operator <<(ostream &os, Employee &e){
    int grosspay = e.basicpay + e.allowance;
    os << "\n";
    os.width(20);
    os<<e.name;
    os.width(8);
    os<<e.basicpay;
    os.width(12);
    os<<e.allowance;
    os.width(8);
    os << grosspay;
    return (os);
```

```
}
int main(){
    Employee e1("Bataa", 10, 10);
    cout<<e1;
    Employee e2("Dorj",20,13);
    cout<<e2;
    return 0;
}
```

```
              Bataa      10          10      20
               Dorj      20          13      33
bye bye Dorj

bye bye Bataa
```

4. Хэрэгжүүлэлт

4.1) Ажилтны классын нэр болон албан тушаалыг хаяган хувьсагч болгон, байгуулагчийг өөрчлөв.

```
class Employee
{

    int id;
    char *name;
    char * position;
    float worked_hours;
    float hourly_salary;
    float bossSalary()
    {
        // boss iin bonus iig bodoj olno.
        return worked_hours * hourly_salary * 1.5;
    }

public:
    Employee(){
        name = new char[20];
        strcpy(name, "");
        position = new char[10];
        worked_hours = 0;
```

```
        hourly_salary = 0;
    }
    Employee(int id = 0, char *name, char *position, float worked_hours =
0, float hourly_salary = 0)
    {
        this->id = id;
        this->name = new char[strlen(name)+1];
        strcpy(this->name,name);

        this->position = new char[strlen(position)+1];
        stpcpy(this->position, position);

        this->worked_hours = worked_hours;
        this->hourly_salary = hourly_salary;
    }
    ~Employee()
    {
        std::cout << "\n the object with " << id << " id  is
destroyed.\n";
        delete this->name;
        delete this->position;
    }
```

4.2) Ажилтны класст “copy()” функц зарлан, нэрээр эрэмбэлэхдээ ашиглав

```
void copy(Employee &e){
        if(name!=NULL){
            delete [] name;
        }
        name = new char[strlen(e.name)+1];
        strcpy(name,e.name);

        position = new char[strlen(e.position)+1];
        strcpy(position, e.position);

        id = e.id;
        worked_hours = e.worked_hours;
        hourly_salary = e.hourly_salary;
    }
```

```
for (int i = 0; i < v.size(); i++)
            {
                int j = i;
                while (j > 0 && strcmp(v[j].getName(), v[j - 1].getName())
< 0)
                {
                    Employee temp;
                    temp.copy(v[j]);
                    v[j] = v[j - 1];
                    v[j - 1].copy(temp);
                    j--;
                }
            }
            cout << "Sorted!!!" << endl;
```

4.3) Шинэ ажилтан үүсгэхдээ ID давхцаж байгаа эсэхийг шалгадаг болгов

```
Employee e1;
            e1.setInfo();
            bool isRep;
            for (int i = 0; i < v.size(); i++)
            {
                if (v[i].getID() == e1.getID())
                {
                    cout << "invalid id!!" << endl;
                    isRep = true;
                    break;
                }
            }
            if (!isRep)
            {
                v.push_back(e1);
                cout << "Added successfully" << endl;
            }
```

4.4) Объектын хаяган хүснэгт үүсгээд нэрээр нь эрэмбэлэв.

```
typedef Employee *Pointer;
    Pointer *employees = new Pointer[v.size()];
    for (int i = 0; i < v.size(); ++i)
    {
        employees[i] = &v[i];
    }
    for (int i = 0; i < v.size(); i++)
    {
        int j = i;
        while (j > 0 && strcmp(employees[j]->getName(), employees[j -
1]->getName()) < 0)
        {
            Employee *temp;
            temp = employees[j];
            employees[j] = employees[j - 1];
            employees[j - 1] = temp;
            j--;
        }
    }
    cout << "Sorted!!!" << endl;
```

5.Дүгнэлт

Хуулагч байгуулагч ашигласнаар шинээр объект үүсгэх болгонд гарааны утга оноох үйлдлийг хялбарчлан, өмнөх оршин буй объектуудаас шинэ объектыг гарган авах боломжтой болж байна. Мөн лабораторийн жишээн дээр гарч байгаагаар класс дотор тэмдэгт мөрийн хаяган хувьсагч ашигласнаар тухайн тэмдэгт мөрийн хэмжээ динамик болох боломжтой болов. Мөн утга оноох үйлдлээс илүү хуулбарлах функц ашигласнаар санах ойн цоорхой үүсэхээс сэргийлж болж байна.

6.Ашигласан материал

1. “Объект хандлагат технологийн C++ програмчлал” Ж.Пүрэв 2008, Улаанбаатар
2. “The C++ programming language” Bjarne Stroustrup 2013, United States
3. https://www.geeksforgeeks.org/copy-constructor-in-cpp/

7.Хавсралт

Employee.h

```
#include <iostream>
#include <string.h>
using namespace std;

class Ant
{
private:
    char name[20];
    float length;
    float width;

public:
    Ant(char n[] = "Ant Bataa", float length = 10, float width=1)
    {
        strcpy(name, n);
        this->length = length;
        this->width = width;
    }
    ~Ant()
    {
        cout << "Ant has retired." << endl;
    }
    void display(){
        cout<<"name: "<<name<<" length: "<<length<<" width:
"<<width<<endl;
    }
};

int main()
{
    typedef Ant *AntPtr;
    int num_ants;
    cout<<"Enter number of ants: ";cin>>num_ants;
    AntPtr *ants = new AntPtr[num_ants];
    for (int i = 0; i < num_ants; ++i)
    {
        ants[i] = new Ant();
```

```
    }

    for (int i = 0; i < num_ants; ++i)
    {
        ants[i]->display();
    }

    return 0;
}
```

Main.cpp

```
#include "Employee.h"
using namespace std;

int main()
{
    cout << "Human resource program by Tuvshin-Erdene 2025" << endl;
    vector<Employee> v;

    while (1)
    {
        int choice;
        cout << "1. list of employees 2. choose employee 3.new employee
4.sort by salary 5.sort by name -1 to exit: ";
        cin >> choice;

        if (choice == 1)
        {
            for (int i = 0; i < v.size(); i++)
            {
                cout << i << "." << endl;
                v[i].getInfo();
            }
            cout << "------------" << endl;
            cout << "total number of employees: " << v.size() << endl;
        }

        else if (choice == 2)
        {
            cout << "Enter the id: ";
```

```
            int cID, num;
            cin >> cID;
            for (int i = 0; i < v.size(); i++)
            {
                if (v[i].getID() == cID)
                {
                    num = i;
                    break;
                }
            }
            v[num].getInfo();
            cout << endl;
            int choice2;
            while (1)
            {
                cout << "1.set information 2.calculate salary 3.increase
work hour -1 to exit: ";
                cin >> choice2;
                if (choice2 == 1)
                {
                    v[num].setInfo();
                }
                else if (choice2 == 2)
                {
                    float salary = v[num].calcSalary();
                    cout << "salary = " << salary << endl;
                }
                else if (choice2 == 3)
                {
                    cout << "enter hour: ";
                    int time;
                    cin >> time;
                    v[num].inc_hour(time);
                }

                else
                {
                    cout << "exiting employee with id : " << cID << endl;
                    break;
                }
```

```
            }
        }
        else if (choice == 3)
        {
            Employee e1;
            e1.setInfo();
            bool isRep;
            for (int i = 0; i < v.size(); i++)
            {
                if (v[i].getID() == e1.getID())
                {
                    cout << "invalid id!!" << endl;
                    isRep = true;
                    break;
                }
            }
            if (!isRep)
            {
                v.push_back(e1);
                cout << "Added successfully" << endl;
            }
        }
        else if (choice == 4)
        {

            for (int i = 0; i < v.size(); i++)
            {
                int j = i;
                while (j > 0 && v[j].calcSalary() > v[j - 1].calcSalary())
                {
                    Employee temp;
                    temp.copy(v[j]);
                    v[j] = v[j - 1];
                    v[j - 1].copy(temp);
                    j--;
                }
            }
            cout << "Sorted!!!" << endl;
        }
        else if (choice == 5)
```

```
        {
            for (int i = 0; i < v.size(); i++)
            {
                int j = i;
                while (j > 0 && strcmp(v[j].getName(), v[j - 1].getName())
< 0)
                {
                    Employee temp;
                    temp.copy(v[j]);
                    v[j] = v[j - 1];
                    v[j - 1].copy(temp);
                    j--;
                }
            }
            cout << "Sorted!!!" << endl;
        }
        else
        {
            break;
        }
    }
    typedef Employee *Pointer;
    Pointer *employees = new Pointer[v.size()];
    for (int i = 0; i < v.size(); ++i)
    {
        employees[i] = &v[i];
    }
    for (int i = 0; i < v.size(); i++)
    {
        int j = i;
        while (j > 0 && strcmp(employees[j]->getName(), employees[j -
1]->getName()) < 0)
        {
            Employee *temp;
            temp = employees[j];
            employees[j] = employees[j - 1];
            employees[j - 1] = temp;
            j--;
        }
    }
```

```
    cout << "Sorted!!!" << endl;
    return 0;
}
```