

Классыг Загварчлан Удамшуулах, Эх Класс Болон Удамшсан Классуудын Харилцаа (Лаборатори №7)

Н. Энхболд

Програм хангамжийн 2-р түвшний оюутан, 20B1NUM0102

1. ОРШИЛ/УДИРТГАЛ

Объект хандалтат програмчлалын удамшил гэх ойлголтыг TwoDimensionalShape класс дээр шаталсан хэлбэрээр үзүүлсэн ба үүнээс Circle, Square, Triangle зэрэг хүүхэд классуудыг бий болгов. Удамшлын горимуудын түлхүүр үгүүдийг ашиглан Эх классын гишүүд рүү хандах Хүүхэд классын харилцааг харуулсан бөгөөд үүн дээр нэмэн удамшин шинээр үүссэн классын байгуулагч функц хэрхэн ажиллаж буйг хавсаргав.

2. ЗОРИЛГО

Shape классыг удамшуулан TwoDimensionalShape гэх классыг тодорхойлох бөгөөд энэхүү классаа дахин удамшуулан Circle, Square, Triangle зэрэг классуудыг тодорхойлон судалгааг хийх:

1. Удамшил
2. Удамшлын төрлүүд
3. Удамшлын давуу тал зэргийг тодорхойлон судлах зорилго тавив.

3. ОНОЛЫН СУДАЛГАА

3.1 Удамшил

Програмчлалын хэлүүдэд удамшуулах гэдэг ойлголт нь хэрэглэсэн зүйлээ дахин өөр зүйлд зориулан өргөтгөж хэрэглэх үед бий болдог. Бид ямар нэгэн зүйлийн ерөнхий шинжээр класс үүсгэн тухайн классаа илүү олон өгөгдөл бүхий тодорхой 1 зүйлд зориулан хэрэглэхдээ тухайн классаа удамшуулж шинэ класс үүсгэдэг. Удамшсан классыг “Эх класс” ба шинээр үүсч буй классыг “Хүүхэд класс” гэж нэрлэдэг.

3.2 Удамшлийн горимууд

Объектыг удамшуулахын тулд өмнөх эх классын гишүүн өгөгдлүүдийг/функциудын “Хүүхэд классаас” хандах горимыг тодорхойлж өгдөг бөгөөд үүнийг удамшлын горим гэдэг. Эдгээр нь: public, private, protected зэрэг түлхүүр үгсээр зохицуулагдаж өгдөг.

- Public: Классыг энэхүү удамшлын горимоор удамшуулж өгснөөр “Хүүхэд класс”-аас эх класст буй гишүүдэд ямар нэгэн хандалтын горимын нэмэлт өөрчлөлтгүй шууд байгаа горимоор нь хандаж болдог.
 - Private гишүүдэд мэдээж шууд хандаж болохгүй
 - Public гишүүдэд шууд хандаж болно
 - Protected гишүүдэд зөвхөн “Хүүхэд класс”-аас шууд хандах боломжтой
- Private: Энэхүү горимоор удамшуулсан “Хүүхэд класс” “Эх класс” руугаа шууд хандах боломжгүй байдаг бөгөөд бүх гишүүд нь хандах боломжгүй private хэлбэртэй болж удамшдаг.
- Protected: Энэхүү горим нь харьцангуй уян хатан буюу гаднаас хандах шууд боломжгүй хэдий ч хүүхэд классаас хандах боломжийг бий болгодог
 - Public болон Protected гишүүдэд шууд хандах боломжтой
 - Private гишүүн рүү ямар ч горимоор удамшуулсан шууд хандах боломжгүй

3.3 Удамшлийн давуу болон сул тал

Давуу талууд:

1. Удамшлыг ашигласнаар ерөнхий ойлголт бүхий классыг элементарчлан шинэ класс үүсгэн өмнөх классын мэдээллийг өөрчлөлгүйгээр дахин ашиглах боломжийг бий болгодог.
2. Тухайн эх классаас үүдэлтэй олон хүүхэд класс үүсгэх бол удамшлыг ашиглан илүү бага кодын тусламжтайгаар бичих мөн ямар нэгэн эх классаас хамаарсан алдаа харьцангуй багасна.
3. Илүү бага зардлаар буюу энэхүү классаа ашиглан илүү бүтээмжийг ихэсгэх боломжтой.

Сул:

1. Хязгаарлагдмал буюу бид эх класст буй өгөгдлийг ашиглах шаардлага гараагүй үед энэ нь илүүц зардлыг бий болгоно.
2. Хэрэв бид олон хүүхэд класс үүсгэх шаардлагагүй тохиолдолд энэхүү эх классыг бий болгох нь ямар ч үр ашиггүй юм.

3.4 Удамшлын төрлүүд

Удамшлын 5 төрөл байдаг.

1. Нэг-нэг удамшил (Энгийн)

- Ямар нэгэн хүүхэд класс зөвхөн 1 эх классаас удамших хэлбэр

```
Class Hvn{...};  
Class Oyutan: public Hvn{...}
```

2. Нэг-олон удамшил (Нийлмэл)

- Хүүхэд класс нь олон эх классаас удамшиж буй хэлбэр

```
Class Aav{...};  
Class Eej{...};  
Class Hvv : public Aav, public Eej{...}
```

3. Олон түвшинт удамшил

- Ямар нэгэн хүүхэд класс удамшин шинээр хүүхэд класс үүсгэх буюу давхар удамших хэлбэр

```
Class Mashin{...};  
Class UraldaaniiMashin : public Mashin{..};  
Class Bugatti: public UraldaaniiMashin{...};
```

4. Шаталсан удамшил

- Нэг эх классаас олон хүүхэд класс үүсэх хэлбэр

```
Class Shape{...};  
Class 2DShape : public Shape{...};  
Class 3DShape : public Shape{...};
```

5. Холимог удамшил

- Энгийн болон шаталсан зэрэг удамшуулах хэлбэрийн нийлбэр хэлбэр

```
Class Shape{..};  
Class 2DShape : public Shape{..};  
Class 3DShape : public Shape{...};  
Class Ogtorgui : public 2DShape, public 3DShape{...};
```

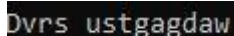
4. ХЭРЭГЖҮҮЛЭЛТ

1. Shape классаас TwoDimensionalShape, цаашлан шаталсан хэлбэрээр Circle, Square, Triangle зэрэг хүүхэд классуудыг үүсгэв. Shape эх класст эдгээр хүүхэд класс бүрт байх name өгөгдлийг тодорхойлсон бөгөөд үүнийг динамик санах ойн new оператор ашиглан тохирсон хэмжээ бүхий санах ой нөөцлөх байдлаар загварчлав.

```
protected:
    char *name;
Shape(char *ner){
    name = new char[strlen(ner) + 1];
    strcpy(name, ner);
    cout << "Dvrs vvssen" << endl << endl;
}
~Shape(){
    delete name;
    cout << "Dvrs ustgagdaw" << endl;
}
```



```
Dvrs vvssen
```



```
Dvrs ustgagdaw
```

2. TwoDimensionalShape класст оройн цэгийн координатууд болон радиусыг гишүүн өгөгдөл болгон авч түүнээс удамшуулан дүрс бүрт харгалзах бусад оройн координатуудыг олов.

Квадрат:

```
private:
    int x1, x2, x3, y1, y2, y3;
public:
    . Square(char *ner, int a, int b, int urt) : TwoDimensionalShape(ner,
a, b, urt){
        x1 = x + r;
        y1 = y;
        x2 = x + r;
```

```

        y2 = y + r;
        x3 = x;
        y3 = y + r;
    }

```

Зөв гурвалжин:

```
private:
```

```
    float x1, x2, y1, y2;
```

```
public:
```

```

    Triangle(char *ner, int a, int b, int urt):TwoDimensionalShape(ner,
a, b, urt){
        y1 = r * cos(30 * PI /180.00) + y;
        y2 = r * cos(30 * PI /180.00) + y;
        x1 = r * sin(30 * PI/180.00) + x;
        x2 = x - r * sin(30 * PI/180.00);
    }

```

3. Constructor функцийг Эх класс болон Хүүхэд класст хоёуланд тодорхойлж өгсөн бөгөөд Хүүхэд классын Constructor нь Эх классын параметрт байгуулагчийг ашиглан гишүүн өгөгдөлдөө утга оноодог байдлаар загварчлав.

```

Dvrs vvssen
Ner: toirog
Toingiin tuw: 10 10
Radius: 5
Talbai: 78.5397
Urt: 31.4159

Dvrs vvssen
Ner: Kvadrat
Koordinatuud:
a(10, 10) b(12, 10)
c(12, 12) d(10, 12)
Urt: 2
Talbai: 4
Perimeter: 8

Dvrs vvssen
Ner: Gurvaljin
Koordinatuud:
a(10, 10) b(14, 16.9282) c(6, 16.9282)
Urt: 8
Talbai: 27.7128
Perimeter: 24

Dvrs ustgagdaw
Dvrs ustgagdaw
Dvrs ustgagdaw
-----

```

5. ДҮГНЭЛТ

Хүүхэд буюу үүсч буй класст байгуулагч тодорхойлон main() функц дотроо объект бий болгон ажлуулж үзэхэд тухайн хүүхэд классын эх класст буй байгуулагч давхар ажиллах бөгөөд энэхүү байгуулагч нь өөрийн гишүүн өгөгдлүүдээ үүсгэж байна. Мөн энэхүү 2 эх класс болон хүүхэд классын байгуулагчдын constructor-ууд дуудагдахад эх классын байгуулагч түрүүлж ажилласан бөгөөд өөрийн гишүүн өгөгдлүүдээ утга оноосны дараачаар хүүхэд классын байгуулагч ажиллаж эхлэв.

6. АШИГЛАСАН МАТЕРИАЛ

1. Объект хандлагат технологийн C++ програмчлал, Ж.Пүрэв, 2008, Улаанбаатар.
2. Объект хандалтат програмчлал – Лекц 08 - Удамшил - https://drive.google.com/file/d/1_AhR0fH2RVLExpzUcGCOHrp4p0km1uyB/view?usp=sharing

7. ХАВСРАЛТ

```

#include <iostream>
#include <math.h>
#define PI 3.14159

using namespace std;
class Shape{
protected:
    char *name;
public:
    void setName(char *ner){
        name = new char[strlen(ner) + 1];
        strcpy(name, ner);
    }
    Shape(char *ner){
        name = new char[strlen(ner) + 1];
        strcpy(name, ner);
        cout << "Dvrs vvssen" << endl << endl;
    }
    Shape(){
        cout << "Dvrs vvssen" << endl << endl;
    }
    ~Shape(){
        delete name;
        cout << "Dvrs ustgagdaw" << endl;
    }
};

class TwoDimensionalShape : public Shape{
protected:
    int x;
    int y;
    int r;
public:
    TwoDimensionalShape(char *ner, int a, int b, int urt) : Shape(ner){
        x = a;
        y = b;
        r = urt;
    }
    TwoDimensionalShape(char *ner) : Shape(ner){
    }
    int getRadius(){
        return r;
    }
};

class Circle : public TwoDimensionalShape{
public:
    float findArea(){
        return PI * r * r;
    }
}

```

```

float findCircle(){
    return 2 * PI * r;
}
Circle() : TwoDimensionalShape("Dugui"){
    x = 0;
    y = 0;
}
Circle(char *ner, int a, int b, int urt) : TwoDimensionalShape(ner,
a, b, urt){
}
void setRadius(float a){
    r = a;
}
void print(){
    cout <<"Ner: " << name << endl;
    cout << "Toirgiin tuw: " << x << " " << y << endl;
    cout << "Radius: " << r << " " << endl;
    cout << "Talbai: " << findArea() << endl;
    cout << "Urt: " << findCircle() << endl << endl;
}
};

```

```

class Square : public TwoDimensionalShape{
private:
    int x1, x2, x3, y1, y2, y3;
public:
    float findArea(){
        return r * r;
    }
    float findFrame(){
        return 4 * r;
    }
    Square() : TwoDimensionalShape("Kvadrat"){
        x = 0;
        y = 0;
        r = 1;
        x1 = x + r;
        y1 = y;
        x2 = x + r;
        y2 = y + r;
        x3 = x;
        y3 = y + r;
    }
    Square(char *ner, int a, int b, int urt) : TwoDimensionalShape(ner,
a, b, urt){
        x1 = x + r;
        y1 = y;
        x2 = x + r;
        y2 = y + r;
        x3 = x;

```



```

        y3 = y + r;
    }
    void setLength(float l){
        r = l;
    }
    void setA(float a, float b){
        x = a;
        y = b;
        x1 = x + r;
        y1 = y;
        x2 = x + r;
        y2 = y + r;
        x3 = x;
        y3 = y + r;
    }
    void print(){
        cout <<"Ner: " << name << endl;
        cout << "Koordinatuud: " << endl;
        cout << "a(" << x << ", " << y << ") ";
        cout << "b(" << x1 << ", " << y1 << ") " << endl;
        cout << "c(" << x2 << ", " << y2 << ") ";
        cout << "d(" << x3 << ", " << y3 << ") " << endl;
        cout << "Urt: " << r << " " << endl;
        cout << "Talbai: " << findArea() << endl;
        cout << "Perimeter: " << findFrame() << endl << endl;
    }
};

class Triangle : public TwoDimensionalShape{
private:
    float x1, x2, y1, y2;
public:
    float findArea(){
        return (r * r * sqrt(3) /2 /2);
    }
    float findFrame(){
        return 3 * r;
    }
    Triangle() : TwoDimensionalShape("Gurvaljin"){
    }
    Triangle(char *ner, int a, int b, int urt) :
TwoDimensionalShape(ner, a, b, urt){
        y1 = r * cos(30 * PI /180.00) + y;
        y2 = r * cos(30 * PI /180.00) + y;
        x1 = r * sin(30 * PI/180.00) + x;
        x2 = x - r * sin(30 * PI/180.00);
    }
    void setLength(float l){
        r = l;
    }
    void setA(float a, float b){
        x = a;

```

```

        y = b;
        y1 = r * cos(30 * PI /180.00) + y;
        y2 = r * cos(30 * PI /180.00) + y;
        x1 = r * sin(30 * PI/180.00) + x;
        x2 = x - r * sin(30 * PI/180.00);
    }
    void print(){
        cout <<"Ner: " << name << endl;
        cout << "Koordinatuud: " << endl;
        cout << "a(" << x << ", " << y << ") ";
        cout << "b(" << x1 << ", " << y1 << ") ";
        cout << "c(" << x2 << ", " << y2 << ") " << endl;
        cout << "Urt: " << r << " " << endl;
        cout << "Talbai: " << findArea() << endl;
        cout << "Perimeter: " << findFrame() << endl << endl;
    }
};

int main(){

    //Toirog
    Circle a("toirog", 10, 10, 5);
    a.print();
    //Kvadrat
    Square s("Kvadrat", 10, 10, 2);
    s.print();
    //Gurvaljin
    Triangle c("Gurvaljin", 10, 10, 8);
    c.print();
}

```