# Жинхэнэ Хийсвэр Функц Тодорхойлох Лабораторийн Ажил (Лаборатори №9)

М.Золжаргал

ХШУИС, МКУТ-ийн багш, zoljargal@num.edu.mn

# 1. ОРШИЛ/УДИРТГАЛ

Дүрс гэдэг хийсвэр класс тодорхойлж түүнээс класс удамшуулсан. Тойрог, дөрвөлжин, гурвалжин классуудыг тодорхойлж, тэдгээрийн хэд хэдэн объект үүсгэн талбайг олж эрэмбэлсэн.

# 2. ЗОРИЛГО

Дүрс классыг тодорхойлж түүнээс удамшсан классуудыг тодорхойлсон. Үүний тулд дараах зорилтуудыг тавьж ажилласан:

1. Дүрс классын гишүүдийг тодорхойлох,
2. Пириметр олдог жинхэнэ хийсвэр функц зарлах,
3. Хоёр хэмжээст дүрс классыг дүрс классаас удамшуулах,
4. Талбай олдог функцыг жинхэнэ хийсвэрээр зарлах
5. Төрөл бүрийн дүрсийн классыг хоёр хэмжээстээс удамшуулах.

# 3. ОНОЛЫН СУДАЛГАА

## 3.1 Хийсвэр функц

Эх классын ямарваа нэг функцыг удамшсан классд өөрөөр тодорхойлж ашиглах боломжийг хийсвэр функц олгоно. Хийсвэр функцыг virtual түлхүүр үгээр тодорхойлно. байгуулагч функцыг хйисвэрээр зарлаж болдоггүй.

Хийсвэр функцыг тодорхойлохдоо:

**virtual** int sum(int a, int b) {

return a+b;

}

## 3.2 Жинхэнэ хийсвэр функц

Аливаа классд заавал байх ёстой мөн удамшсан тохиолдолд охин классд дахин програмчлагдаж ашиглагдах функцуудыг жинхэнэ хийсвэр функц гэнэ. Энэ нь хэрвээ уг классаас ямар нэг класс удамшвал тухайн охин классд тодорхойлогдсон байх шаардлагатай функцуудыг жинхэнэ хийсвэрээр зарлана гэсэн үг.

Жинхэнэ хийсвэр функц зарлахдаа:

**virtual** int sum(int a, int b) = 0;

## 3.3 Функц дахин програмчлах

Эх классын хийсвэрээр тодорхойлсон функцыг охин классд дахин өөр дүрмээр тодорхойлж ашиглахыг функц дахин програмчлах гэнэ. Хийсвэрээр тодорхойлоогүй функцыг дахин програмчлахгүй. Жишээ нь дүрс болгоны талбайг өөр аргаар олдог, тйимээс талбай олох функцыг дүрс болгоны классд өөрөөр тодорхойлох шаардлагатай болно.

## 3.4 Хийсвэр класс

Жинхэнэ хийсвэр класс агуулсан классыг хийсвэр класс гэнэ. Хийсвэр классаас объект үүсдэггүй.

## 3.5 Удамшил ба байгуулагч (устгагч)

Удамшсан классын объектыг байгуулах үед эх классуудын анхдагч байгуулагч функцууд дуудагдаж ажилладаг. Энэ нь нэг гишүүнд хэд хэдэн удаа утга онооход хүргэнэ. Тиймээс удамших классын байгуулагч эх классын байгуулагчийн хийснийг давтаж хийхгүйгээр зохицуулах шаардлагатай болоод байна. Энэ асуудлыг дараах байдлаар шийдвэрлэнэ:

Engineer::Engineer(char\* name, int salary, int allowance) : employee(name, salary, allowance) {

cout << “Eh klassiin baiguulagchaar damjuulj utga onoogdloo”;

}

Харин устгагч функцын хувьд эхлээд удамшсан функцын устгагч ажиллаад дараа нь эх классын устгагч ажилладаг.

# 4. ХЭРЭГЖҮҮЛЭЛТ

Дүрс классыг тодорхойлсон

class Shape {

protected:

    char name[20];

public:

    Shape(char a[]="durs") {

        strcpy(name, a);

    }

    ~Shape() {

        cout << "ustlaa\n";

    }

    void setName(char\* a) {

        strcpy(name, a);

    }

    char\* getName() {

        return name;

    }

    virtual float getPirimetr() = 0;

};

хавтгай дүрс классыг тодорхойлсон

class TwoDShape:public Shape {

public:

    TwoDShape(char a[]="2\_hemjeest\_durs"):Shape(a) {

    }

    virtual float getArea() = 0;

};

түүний дараа тойрог, дөрвөлжин, гурвалжин классуудыг тодорхойлсон

class Circle:public TwoDShape {

protected:

    float o\_x, o\_y, radius;

public:

    Circle(char a[]="toirog", float b=0, float c=0, float d=0):TwoDShape(a) {

        strcpy(name, a);

        o\_x = b;

        o\_y = c;

        radius = d>=0 ? d : 0;

    }

    void setCoordinateO(float a, float b) {

        o\_x = a;

        o\_y = b;

    }

    float getX() {

        return o\_x;

    }

    float getY() {

        return o\_y;

    }

    float getPirimetr() {

        return 2\*M\_PI\*radius;

    }

    float getArea() {

        return M\_PI\*radius\*radius;

    }

    void setRadius(float a) {

        radius = a>=0 ? a : radius;

    }

    float getRadius() {

        return radius;

    }

};

class Square:public TwoDShape {

protected:

    float o1x, o1y, o2x, o2y, o3x, o3y, o4x, o4y, sideLenght;

public:

    Square(char a[]="Durvuljin", float b=0, float c=0, float d=0) {

        strcpy(name,a);

        sideLenght = d>=0 ? d : 0;

        o1x = b;

        o1y = c;

        o2x = b+d;

        o2y = c;

        o3x = b;

        o3y = c-d;

        o4x = b+d;

        o4y = c-d;

    }

    void setCoordinateO1(float a, float b) {

        o1x = a;

        o1y = b;

        o2x = a+sideLenght;

        o2y = b;

        o3x = a;

        o3y = b-sideLenght;

        o4x = a+sideLenght;

        o4y = b-sideLenght;

    }

    float geto1x() {

        return o1x;

    }

    float geto1y() {

        return o1y;

    }

    float getPirimetr() {

        return 4\*sideLenght;

    }

    float getArea() {

        return sideLenght\*sideLenght;

    }

    void setSideLenght(float a) {

        sideLenght = a>=0 ? a : sideLenght;

    }

    float getSideLenght() {

        return sideLenght;

    }

};

class Triangle:public TwoDShape {

protected:

    float o1x, o1y, o2x, o2y, o3x, o3y, sideLenght;

public:

    Triangle(char a[]="gurvaljin", float b=0, float c=0, float d=0) {

        strcpy(name, a);

        sideLenght = d>=0 ? d : 0;

        o1x = b;

        o1y = c;

        o2x = b - d/2;

        o2y = c - sideLenght\*sin(M\_PI/3);

        o3x = b + d/2;

        o3y = c - sideLenght\*sin(M\_PI/3);

    }

    void setCoordinateo1(float a, float b) {

        o1x = a;

        o1y = b;

        o2x = a - sideLenght/2;

        o2y = b - sideLenght\*sin(M\_PI/3);

        o3x = a + sideLenght/2;

        o3y = b - sideLenght\*sin(M\_PI/3);

    }

    float geto1x() {

        return o1x;

    }

    float geto1y() {

        return o1y;

    }

    float getPirimetr() {

        return sideLenght\*3;

    }

    float getArea() {

        return sideLenght\*sideLenght\*sin(M\_PI/3)/2;

    }

    void setSideLenght(float a) {

        sideLenght = a>=0 ? a : sideLenght;

    }

    float getSideLenght() {

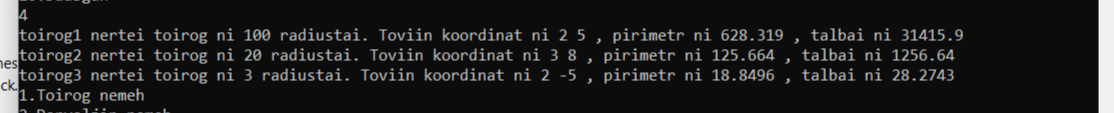
        return sideLenght;

    }

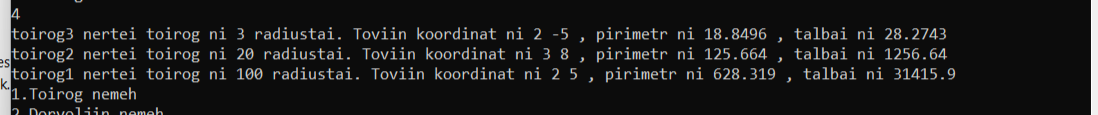
};

тэгээд дүрс болгоны мэдээллийг харж болдог мөн талбайгаар нь эрэмбэлж болно.

дүрсүүдийн мэдээллийг харуулж байгаа байдал:



талбайгаар нь эрэмбэлсний дараах байдал:



# 5. ДҮГНЭЛТ

Удамшсан классд ямар ямар классыг заавал тодорхойлсон байж объект байгуулах боломжтой вэ гэдгийг жинхэнэ хийсвэр функц ашиглан харуулна. Мөн нэг гишүүн өгөгдөлд олон удаа утга оноохоос сэргийлж байгуулагч функцыг зарлахдаа эх классын байгуулагчид утгыг дамжуулж оноох нь програмыг хурдан ажиллахад тусална.

# 6. АШИГЛАСАН МАТЕРИАЛ

1. <https://www.geeksforgeeks.org/virtual-base-class-in-c/>

2. <https://www.geeksforgeeks.org/virtual-function-cpp/>

# 7. ХАВСРАЛТ

#include <iostream>

#include <string.h>

#include <math.h>

#include <vector>

using namespace std;

class Shape {

protected:

    char name[20];

public:

    Shape(char a[]="durs") {

        strcpy(name, a);

    }

    ~Shape() {

        cout << "ustlaa\n";

    }

    void setName(char\* a) {

        strcpy(name, a);

    }

    char\* getName() {

        return name;

    }

    virtual float getPirimetr() = 0;

};

class TwoDShape:public Shape {

public:

    TwoDShape(char a[]="2\_hemjeest\_durs"):Shape(a) {

    }

    virtual float getArea() = 0;

};

class Circle:public TwoDShape {

protected:

    float o\_x, o\_y, radius;

public:

    Circle(char a[]="toirog", float b=0, float c=0, float d=0):TwoDShape(a) {

        strcpy(name, a);

        o\_x = b;

        o\_y = c;

        radius = d>=0 ? d : 0;

    }

    void setCoordinateO(float a, float b) {

        o\_x = a;

        o\_y = b;

    }

    float getX() {

        return o\_x;

    }

    float getY() {

        return o\_y;

    }

    float getPirimetr() {

        return 2\*M\_PI\*radius;

    }

    float getArea() {

        return M\_PI\*radius\*radius;

    }

    void setRadius(float a) {

        radius = a>=0 ? a : radius;

    }

    float getRadius() {

        return radius;

    }

};

class Square:public TwoDShape {

protected:

    float o1x, o1y, o2x, o2y, o3x, o3y, o4x, o4y, sideLenght;

public:

    Square(char a[]="Durvuljin", float b=0, float c=0, float d=0) {

        strcpy(name,a);

        sideLenght = d>=0 ? d : 0;

        o1x = b;

        o1y = c;

        o2x = b+d;

        o2y = c;

        o3x = b;

        o3y = c-d;

        o4x = b+d;

        o4y = c-d;

    }

    void setCoordinateO1(float a, float b) {

        o1x = a;

        o1y = b;

        o2x = a+sideLenght;

        o2y = b;

        o3x = a;

        o3y = b-sideLenght;

        o4x = a+sideLenght;

        o4y = b-sideLenght;

    }

    float geto1x() {

        return o1x;

    }

    float geto1y() {

        return o1y;

    }

    float getPirimetr() {

        return 4\*sideLenght;

    }

    float getArea() {

        return sideLenght\*sideLenght;

    }

    void setSideLenght(float a) {

        sideLenght = a>=0 ? a : sideLenght;

    }

    float getSideLenght() {

        return sideLenght;

    }

};

class Triangle:public TwoDShape {

protected:

    float o1x, o1y, o2x, o2y, o3x, o3y, sideLenght;

public:

    Triangle(char a[]="gurvaljin", float b=0, float c=0, float d=0) {

        strcpy(name, a);

        sideLenght = d>=0 ? d : 0;

        o1x = b;

        o1y = c;

        o2x = b - d/2;

        o2y = c - sideLenght\*sin(M\_PI/3);

        o3x = b + d/2;

        o3y = c - sideLenght\*sin(M\_PI/3);

    }

    void setCoordinateo1(float a, float b) {

        o1x = a;

        o1y = b;

        o2x = a - sideLenght/2;

        o2y = b - sideLenght\*sin(M\_PI/3);

        o3x = a + sideLenght/2;

        o3y = b - sideLenght\*sin(M\_PI/3);

    }

    float geto1x() {

        return o1x;

    }

    float geto1y() {

        return o1y;

    }

    float getPirimetr() {

        return sideLenght\*3;

    }

    float getArea() {

        return sideLenght\*sideLenght\*sin(M\_PI/3)/2;

    }

    void setSideLenght(float a) {

        sideLenght = a>=0 ? a : sideLenght;

    }

    float getSideLenght() {

        return sideLenght;

    }

};

main() {

    vector<Circle> a;

    vector<Square> b;

    vector<Triangle> c;

    int n;

    float x,y,urt;

    char ner[20];

    while(1) {

        cout << "1.Toirog nemeh\n";

        cout << "2.Dorvoljin nemeh\n";

        cout << "3.Gurvaljin nemeh\n";

        cout << "4.Toirguudiig haruulah\n";

        cout << "5.Dorvoljinguudiig haruulah\n";

        cout << "6.Gurvaljinguudiig haruulah\n";

        cout << "7.Toirguudiig talbaigaar ni erembleh\n";

        cout << "8.Dorvoljinguudiig talbaigaar ni erembleh\n";

        cout << "9.Gurvaljinguudiig talbaigaar ni erembleh\n";

        cout << "10.Duusgah\n";

        cin >> n;

        if(n==1) {

            cout << "Toirgiin neriig oruul:";

            cin >> ner;

            cout << "Toirgiin toviin koordinatiig oruul:";

            cin >> x >> y;

            cout << "Radiusiig oruul:";

            cin >> urt;

            Circle k(ner,x,y,urt);

            a.push\_back(k);

        }

        if(n==2) {

            cout << "Dorvoljnii neriig oruul:";

            cin >> ner;

            cout << "Dorvoljnii zuun deed oroin koordinatiig oruul:";

            cin >> x >> y;

            cout << "Taliin urtiig oruul:";

            cin >> urt;

            Square k(ner,x,y,urt);

            b.push\_back(k);

        }

        if(n==3) {

            cout << "Gurvaljnii neriig oruul:";

            cin >> ner;

            cout << "Gurvaljnii deed oroin koordinatiig oruul:";

            cin >> x >> y;

            cout << "Taliin urtiig oruul:";

            cin >> urt;

            Triangle k(ner,x,y,urt);

            c.push\_back(k);

        }

        if(n==4) {

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

                cout << a[i].getName() << " nertei toirog ni ";

                cout << a[i].getRadius() << " radiustai. Toviin koordinat ni ";

                cout << a[i].getX() << " " << a[i].getY() << " , pirimetr ni ";

                cout << a[i].getPirimetr() << " , talbai ni ";

                cout << a[i].getArea() << endl;

            }

        }

        if(n==5) {

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

                cout << b[i].getName() << " nertei dorvoljin ni ";

                cout << b[i].getSideLenght() << " taliin urttai. Zuun deed oroin koordinat ni ";

                cout << b[i].geto1x() << " " << b[i].geto1y() << " , pirimetr ni ";

                cout << b[i].getPirimetr() << " , talbai ni ";

                cout << b[i].getArea() << endl;

            }

        }

        if(n==6) {

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

                cout << c[i].getName() << " nertei gurvaljin ni ";

                cout << c[i].getSideLenght() << " taliin urttai. Deed oroin koordinat ni ";

                cout << c[i].geto1x() << " " << c[i].geto1y() << " , pirimetr ni ";

                cout << c[i].getPirimetr() << " , talbai ni ";

                cout << c[i].getArea() << endl;

            }

        }

        if(n==7) {

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

                for(int j=1; j<a.size(); j++) {

                    if(a[j-1].getArea() > a[j].getArea()) {

                        swap(a[j-1],a[j]);

                    }

                }

            }

        }

        if(n==8) {

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

                for(int j=1; j<b.size(); j++) {

                    if(b[j-1].getArea() > b[j].getArea()) {

                        swap(b[j-1],b[j]);

                    }

                }

            }

        }

        if(n==9) {

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

                for(int j=1; j<c.size(); j++) {

                    if(c[j-1].getArea() > c[j].getArea()) {

                        swap(c[j-1],c[j]);

                    }

                }

            }

        }

        if(n==10) {

            break;

        }

    }

}