**ЕРӨНХИЙЛӨЛ БА НАРИЙВЧЛАЛ**

**(ЛАБОРАТОРИ №7)**

Э.Уранчимэг

ХШИУС, МКУТ, Программ хангамжийн III түвшний оюутан, 21B1NUM0609@stud.num.edu.mn

1. **ОРШИЛ**

Энэ тайлангийн онолын судалгаа хэсэгт удамшил болон байгуулагч функцийн холбоо хамаарал, ерөнхийлөл, нарийвчлалыг судалж тайлбарласан. Өмнөх хичээл дээр удамшуулсан классын эх классуудад байгуулагч функц тодорхойлж удамшлын үед эх классын параметертэй байгуулагчыг дуудаж ажиллуулсан хэрэгжүүлэлтийг харах боломжтой. Хавсралт хэсгээс дэлгэрэнгүй кодыг унших боломжтой.

1. **ЗОРИЛГО**

Энэ лабораторын хүрээнд удамшил ба байгуулагч функцийн холбоо хамаарлыг бүрэн судалж ойлгох ба бодлогын хүрээнд хэрэгжүүлнэ. Үүний тулд дараах зорилтыг тавьж ажилласан:

1. Онолын судалгаа хийх,
2. Байгуулагч функц тодорхойлох,
3. Объект үүсгэж талбайгаар нь эрэмбэлэх,
4. **ОНОЛЫН СУДАЛГАА**

*3.1 Удамшил ба байгуулагч функцийн хамаарал?*

Хүүхэд классын объект байгуулах үед эхлээд эх классын байгуулагч функц дараа нь удамших классын байгуулагч дуудагддаг. Эх классын анхдагч байгуулагч нь хүүхэд классын байгуулагч ажиллах бүрд дуудагддаг бөгөөд хүүхэд классын байгуулагч эх классынхаа хийснийг давтан хийхгүйгээр зөвхөн түүний зогсох газраас үргэлжлүүлэн ажиллах ёстой.

*3.2 Ерөнхийлөл гэж юу вэ? c++ хэл дээр хэрхэн хэрэгжүүлдэг вэ?*

Олон ижил төстэй классуудын нийтлэг (common) шинжүүдийг агуулсан 1 эх класс үүсгээд өнөөх төстэй классуудыг уг эх классаас удамшуулахыг Generalization гэнэ. Ерөнхийдөө доороос дээш гэсэн чиглэлтэй. Удамшил ашиглан хэрэгжүүлдэг. Жишээ нь багш ажилтан, оюутан классуудад бүгдэд нь нэр, нас, регистрийн дугаар гэсэн гэсэн шинжүүд байгаа бол эдгээрийг агуулсан Хүн гэсэн эх класс үүсгэж ерөнхийллийг хэрэгжүүлнэ.

*3.3 Нарийвчлал гэж юу вэ?*

Нэг эх классын ерөнхий шинжийг нь бодож үүсгээд түүнээс хүүхэд класс үүсгэх. Дээрээс доошоо гэсэн чиглэлтэй.

*3.4 Функц дахин тодорхойлох гэж юу вэ? Жишээгээр тайлбарлах.*

Эх классаас удамшсан гишүүн функцыг хүүхэд класс дотор ахин тодорхойлж болдог ба ижил параметр, ижилхэн нэртэй байх ба дотор нь хийгдэх үйлдэл нь өөр буюу өөр аргаар ахин тодорхойлдог.

Жишээ нь: Аав классаас хүүхэд классын хоол идэх удамшвал аав класс хоол идэх нь халбагаар зүүн гараараа хоол идэхээр байсан ч хүүхэд класст ахин тодорхойлон гараараа идэх гэдэг үйлдлийг хийж болно.

1. **ХЭРЭГЖҮҮЛЭЛТ**

Лаб 6-д хийсэн circle, square, triangle классуудаас олон объект үүсгэж дүрс хооронд талбайгаар нь эрэмбэлсэн.

Circle a("Dugui", 10, 10, 5) ;

Circle b("Dugui", 10, 10, 4) ;

Square c("Kvadrat", 10, 10, 3);

Square d("Kvadrat", 10, 10, 2);

Triangle e("Gurvaljin", 10, 10, 8);

Triangle f("Gurvaljin", 10, 10, 5);

TwoDShape \*shapes[6];

shapes[0]= &a;

shapes[1]= &b;

shapes[2]= &c;

shapes[3]= &d;

shapes[4]= &e;

shapes[5]= &f;

for(int i=0; i<6; i++){

for(int j=i+1; j<6; j++){

if(shapes[i]->findArea()>shapes[j]->findArea()){

shp=shapes[i];

shapes[i]=shapes[j];

shapes[j]=shp;

}

}

}

cout<<"Ur dun: \n";

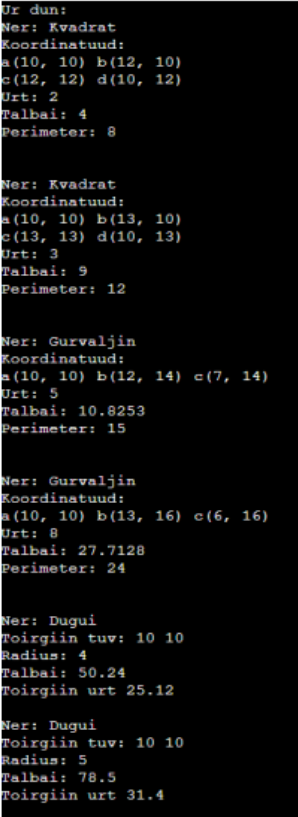
for(int i=0; i<6; i++){

shapes[i]->print();

cout<<endl;

}

Үр дүн нь:



1. **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 өгөгдлүүддээ утга оноосны дараачаар хүүхэд классын байгуулагч ажиллаж эхлэв.

1. **АШИГЛАСАН МАТЕРИАЛ**

* “Объект хандлагат программчлал лекцийн материал”, М.Золжаргал

1. **ХАВСРАЛТ**

Нэгдсэн кодыг хавсаргав.

shape.h

#ifndef \_\_shape\_\_

#define \_\_shape\_\_

#include<iostream>

#include<string.h>

using namespace std;

class Shape {

protected:

char \*name;

public:

void setName(const char \*name);

char \*get\_name();

Shape();

Shape(const char \*name);

~Shape();

};

#endif

shape.cpp

#include "shape.h"

void Shape::setName(const char \*name) {

if (this->name != nullptr) {

delete[] this->name;

}

this->name = new char[strlen(name) + 1];

strcpy(this->name, name);

}

Shape::Shape() {

this->name = new char[20];

strcpy(this->name, "default");

cout << "Durs uusev" << endl;

}

Shape::Shape(const char\* name) {

this->name = new char[strlen(name) + 1];

strcpy(this->name, name);

cout << "Durs(para) uusev" << endl;

}

Shape::~Shape() {

delete[] this->name;

cout << "Durs ustav" << endl;

}

char \*Shape::get\_name() {

return name;

}

twoDshape.h

#ifndef \_\_twoD\_\_

#define \_\_twoD\_\_

#include"shape.h"

class TwoDShape : public Shape {

protected:

int x, y, r ;

float S;

public:

TwoDShape(const char \*ner, int a, int b, int urt);

TwoDShape(const char \*ner);

virtual float findArea()=0;

virtual void print()=0;

};

#endif

twoDshape.cpp

#include"twoDshape.h"

TwoDShape::TwoDShape(const char \*ner, int a, int b, int urt) : Shape(ner){

x = a ;

y = b ;

r = urt ;

}

TwoDShape::TwoDShape(const char \*ner) : Shape(ner){

}

circle.h

#ifndef \_\_Circle\_\_

#define \_\_Circle\_\_

#include"twoDshape.h"

class Circle : public TwoDShape {

public:

float findArea();

float findPerimeter();

Circle();

Circle(const char \*ner, int a, int b, int urt);

void setRadius(float a);

void print();

};

#endif

circle.cpp

#include "circle.h"

#define Pi 3.14

float Circle::findArea(){

S=Pi \* r \* r ;

return S;

}

float Circle::findPerimeter(){

return 2 \* Pi \* r ;

}

Circle::Circle() : TwoDShape("Dugui"){

x = 0;

y = 0;

}

Circle::Circle(const char \*ner, int a, int b, int urt) : TwoDShape(ner, a, b, urt){

}

void Circle::setRadius(float a){

r = a ;

}

void Circle::print(){

cout << "Ner: " << name << endl ;

cout << "Toirgiin tuv: " << x << " " << y << endl ;

cout << "Radius: " << r << endl ;

cout << "Talbai: " << findArea() << endl ;

cout << "Toirgiin urt " << findPerimeter() << endl ;

}

square.h

#ifndef \_\_Square\_\_

#define \_\_Square\_\_

#include"twoDshape.h"

class Square : public TwoDShape {

private:

int x1, x2, x3, y1, y2, y3 ;

public:

float findArea();

float findPerimeter();

Square();

Square(const char \*ner, int a, int b, int urt);

void setLength(float l);

void setA(float a, float b);

void print();

};

#endif

square.cpp

#include "square.h"

float Square::findArea(){

S=r\*r ;

return S;

}

float Square::findPerimeter(){

return 4 \* r ;

}

Square::Square() : TwoDShape("Kvadrat"){

x = 0 ;

y = 0 ;

r = 1;

x1 = x + r ;

y1 = y ;

x2 = x + r ;

y2 = y + r ;

x3 = x ;

y3 = y + r ;

}

Square::Square(const char \*ner, int a, int b, int urt) : TwoDShape(ner, a, b, urt){

x1 = x + r ;

y1 = y ;

x2 = x + r ;

y2 = y + r ;

x3 = x ;

y3 = y + r ;

}

void Square::setLength(float l){

x = l ;

y = l ;

x1 = x + r ;

y1 = y ;

x2 = x + r ;

y2 = y + r ;

x3 = x ;

y3 = y + r ;

}

void Square::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 Square::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: " << findPerimeter() << endl << endl;

}

triangle.h

#ifndef \_\_Triangle\_\_

#define \_\_Triangle\_\_

#include"twoDshape.h"

class Triangle : public TwoDShape {

private:

int x1, x2, y1, y2 ;

public:

float findArea();

float findPerimeter();

Triangle();

Triangle(const char \*ner, int a, int b, int urt);

void setLength(float l);

void setA(float a, float b);

void print();

};

#endif

triangle.cpp

#include"triangle.h"

#include<math.h>

float Triangle::findArea(){

S=(r\*r\*sqrt(3)/2 /2);

return S;

}

float Triangle::findPerimeter(){

return 3 \* r ;

}

Triangle::Triangle() : TwoDShape("Gurvaljin"){

}

Triangle::Triangle(const char \*ner, int a, int b, int urt) : TwoDShape(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 Triangle::setLength(float l){

r = l ;

}

void Triangle::setA(float a, float b){

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 Triangle::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: " << findPerimeter() << endl << endl;

}

main.cpp

#include "circle.h"

#include "triangle.h"

#include "square.h"

#include <iostream>

using namespace std;

int main(){

TwoDShape\* shp;

Circle a("Dugui", 10, 10, 5) ;

Circle b("Dugui", 10, 10, 4) ;

Square c("Kvadrat", 10, 10, 3);

Square d("Kvadrat", 10, 10, 2);

Triangle e("Gurvaljin", 10, 10, 8);

Triangle f("Gurvaljin", 10, 10, 5);

TwoDShape \*shapes[6];

shapes[0]= &a;

shapes[1]= &b;

shapes[2]= &c;

shapes[3]= &d;

shapes[4]= &e;

shapes[5]= &f;

for(int i=0; i<6; i++){

for(int j=i+1; j<6; j++){

if(shapes[i]->findArea()>shapes[j]->findArea()){

shp=shapes[i];

shapes[i]=shapes[j];

shapes[j]=shp;

}

}

}

cout<<"Ur dun: \n";

for(int i=0; i<6; i++){

shapes[i]->print();

cout<<endl;

}

}