МІНІСТЕРТСВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра АПОТ

Звіт

з практичного заняття №3

з дисципліни «Об’єктно-орієнтоване програмування»

# на тему: «СПАДКУВАННЯ. ІЄРАРХІЯ КЛАСІВ. ПОЛІМОРФІЗМ»

Варіант №14/3

Виконав: Перевірила:

ст. групи КИУКИу-21-1

Сургай Р. В.

Малькова І. А.

Харків 2021

МЕТА РОБОТИ: Вивчення реалізації в мові С++ механізмів успадкування та поліморфізму.

ЗАВДАННЯ:

1. *Побудувати ієрархію класів, складовими частинами якої будуть сутності, зазначені в завданні.*

*2 Розробити діаграму класів ієрархії спадкування. Звіт з практичного заняття обов’язково повинен містити UML-діаграму розроблених класів та класу-контейнеру!*

*3. Засобами мови С++ реалізувати ієрархію спадкування. Базовий клас визначити абстрактним.*

*4. В базовому класі визначити вказану для варіанту віртуальну функцію, перевизначити її у похідних класах.*

*6. Створити клас-контейнер, який володіє динамічним списком об'єктів, які є екземплярами похідних класів з ієрархії успадкування.*

*7. Включити в клас-контейнер функцію, яка дозволяє додавати нові елементи в динамічний список об'єктів, та функцію, що обробляє список об'єктів з використанням механізмів поліморфізму.*

*8. Всі класи з ієрархії спадкування повинні включати в себе конструктори за замовчуванням, конструктор копіювання та конструктор переміщення. В клас-контейнер досить включити конструктор за замовчуванням.*

*9. Реалізувати вказану у варіанті функціональність класу-контейнеру.*

*10. У функції main здійснити демонстрацію функціональності розроблених класів.*

Опишіть базовий клас водойомів та похідні класи: озеро, річка, басейн, море, океан. Віртуальна функція:Розрахувати площі та периметри водойомів. Функціональність класу-контейнеру:Знаходження водойому з найбільшою глибиною. Виведення об’єктів контейнеру у порядку зменшення площі. Виведення об’єктів контейнеру, що згруповані за ознакою (солоні та прісні).

ІНТЕРФЕЙС:

#include <iostream>

#include <string>

#include <vector>

using namespace std;

// Базовый класс "Водоемы"

class Water {

public:

Water() {

name = "None";

square = 0;

volume = 0;

maxDepth = 0;

midDepth = 0;

salty = false;

}

Water(string name, int s, int v, int max, int mid, bool sal) {

this->name = name;

this->square = s;

this->volume = v;

this->maxDepth = max;

this->midDepth = mid;

this->salty = sal;

}

// ~Water();

string name; // Название

int square; // Площадь поверхности

int volume; // Объем воды

int maxDepth; // Максимальная глубина

int midDepth; // Средняя глубина

bool salty; // Соленый или престный

virtual void sq() = 0;

};

// Производный класс "Океаны"

class Ocean : public Water {

public:

Ocean() {

worldVolume = 0;

}

Ocean(string name, int s, int v, int max, int mid, bool sal, double w) : Water(name, s, v, max, mid, sal) {

if (w > 100 || w < 0) {

cout << "\*ОШИБКА\*\nЗначение должно иметь параметр от 0% до 100%" << endl;

this->worldVolume = 0;

}

else {

this->worldVolume = w;

}

}

// ~Ocean();

double worldVolume; // Процент объема воды из Мирового океана

void sq() override {

cout << this->name << ". Площадь поверхности равна " << this->square << endl;

}

};

// Производный класс "Моря"

class Sea : public Water {

public:

Sea() {

oceanClass = "None";

borderCountry = "None";

flowingRivers = "None";

coastline = 0;

}

Sea(string name, int s, int v, int max, int mid, bool sal, string oc, string bo, string fl, int co) : Water(name, s, v, max, mid, sal) {

this->oceanClass = oc;

this->borderCountry = bo;

this->flowingRivers = fl;

this->coastline = co;

}

// ~Sea();

string oceanClass; // Пренадлежность к океану

string borderCountry; // Страны, которые омывает

string flowingRivers; // Впадающие реки

int coastline; // Длина береговой линии

void sq() override {

cout << this->name << ". Площадь поверхности равна " << this->square << endl;

}

};

// Производный класс "Озера"

class Lake : public Water {

public:

Lake() {

origin = "None";

country = "None";

heightSea = 0;

}

Lake(string name, int s, int v, int max, int mid, bool sal, string ori, string co, int he) :

Water(name, s, v, max, mid, sal) {

this->origin = ori;

this->country = co;

this->heightSea = he;

}

// ~Lake();

string origin; // Происхождение

string country; // Страна, где находится

int heightSea; // Высота над уровнем моря

bool salty; // Соленое или престное

void sq() override {

cout << this->name << ". Площадь поверхности равна " << this->square << endl;

}

};

// Производный класс "Реки"

class River : public Water {

public:

River() {

supply = "None";

source = "None";

mouth = "None";

heightSource = 0;

lenght = 0;

}

River(string name, int s, int v, int max, int mid, bool sal, string su, string so, string mo, int he, int le) :

Water(name, s, v, max, mid, sal) {

this->supply = su;

this->source = so;

this->mouth = mo;

this->heightSource = he;

this->lenght = le;

}

// ~River();

string supply; // Питание

string source; // Исток

string mouth; // Устье

int heightSource; // Высота истока

int lenght; // Длина

void sq() override {

cout << this->name << ". Площадь бассейна равна " << this->square << endl;

}

};

class Container {

public:

Water\*\* arrayWater;

int waterCounter;

Container() {

arrayWater = new Water \* [0];

waterCounter = 0;

}

// Нахождения самого глубокого водоема

void maxDepthWater() {

Water\* maxDep = arrayWater[0];

for (int i = 0; i < waterCounter - 1; i++) {

if (maxDep->maxDepth < arrayWater[i + 1]->maxDepth) {

maxDep = arrayWater[i + 1];

}

}

cout << maxDep->name << " имеет максимальную глубину: " << maxDep->maxDepth << " метров\n\n";

}

// Сортирорвка водоемов по площади поверхности

void sortWaterSquare() {

int max;

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

max = i; // макс. эл. - первый

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

if (arrayWater[max]->square < arrayWater[j]->square) { // Нахождние макс. эл

max = j;

}

}

if (max != i) { // Если элемент поменялся, делаем замену

swap(arrayWater[i], arrayWater[max]);

}

}

cout << "Перечень водоемов (в порядке убывания) по их площади поверхности:\n";

for (int k = 0; k < waterCounter; k++) {

cout << arrayWater[k]->name << " = " << arrayWater[k]->square << " км^2\n";

}

cout << endl;

}

// Сортировка водоемов на "соленые" и "престные"

void sortSalinity() {

string allsa, allfr;

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

if (arrayWater[i]->salty == true) {

allsa += "<" + arrayWater[i]->name + "> ";

}

else {

allfr += "<" + arrayWater[i]->name + "> ";

}

}

cout << "Соленые водоемы: " << allsa << endl;

cout << "Пресные водоемы: " << allfr << endl << endl;

}

// Метод добавления нового водоема

void addWater(Water\* newWater) {

waterCounter++;

Water\*\* allWater = new Water \* [waterCounter];

for (int i = 0; i < waterCounter - 1; i++) {

allWater[i] = arrayWater[i];

}

allWater[waterCounter - 1] = newWater;

delete[]arrayWater;

arrayWater = allWater;

}

};

ФУНКЦІЯ MAIN():

int main() {

setlocale(LC\_ALL, "ru");

Ocean tihiy = Ocean("Тихий океан", 178684000, 710360000, 11022, 3984, true, 53);

Ocean indian = Ocean("Индийский океан", 76174000, 282650000, 7729, 3711, true, 20);

Sea black = Sea("Черное море", 422000, 555000, 2210, 1240, true, "Атлантический океан",

"Россия, Украина, Румыния, Болгария, Турция, Грузия, Абхазия",

"Дунай, Днепр, Днестр", 3400);

Sea red = Sea("Красное море", 438000, 233000, 2211, 490, true, "Индийский океан",

"Египет, Судан, Джибути, Эритрей, Йемен, Израиль, Иордания",

"None", 4600);

Lake baikal = Lake("Озеро Байкал", 31722, 23615, 1642, 744, false, "Тектоническое",

"Россия", 456);

Lake vika = Lake("Озеро Виктория", 68000, 2760, 80, 40, false, "Тектоническое",

"Танзания, Кения, Уганда", 1134);

River nil = River("Река Нил", 3349000, NULL, 11, 7, false, "Дождевое питание", "Озеро Виктория",

"Средиземное море", 1134, 6854);

River amazon = River("Река Амазонка", 7180000, 7280, 100, 50, false, "Дождевое питание",

"Реки Укаяли и Мараньон", "Атлантический океан", 110, 6500);

Container con;

con.addWater(&tihiy);

con.addWater(&indian);

con.addWater(&black);

con.addWater(&red);

con.addWater(&baikal);

con.addWater(&vika);

con.addWater(&nil);

con.addWater(&amazon);

con.maxDepthWater(); // Нахождение максимальной глубины

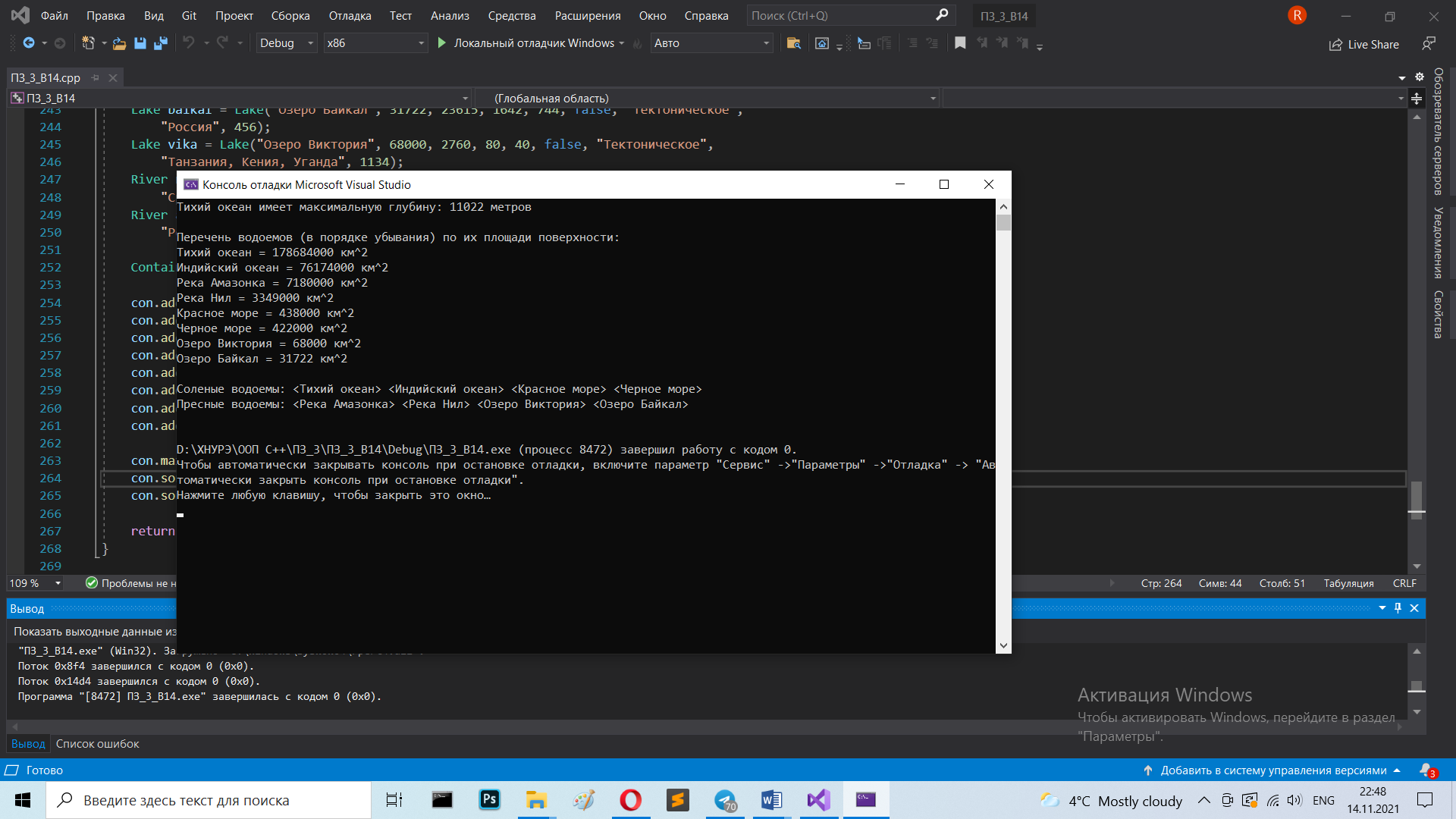
con.sortWaterSquare(); // Сортировка по площади

con.sortSalinity(); // Разбиение на "соленые" и "престные"

return 0;

}

РЕЗУЛЬТАТ:



UML-діаграма класів:

|  |
| --- |
| Water |
| # name  # square  # volume  # maxDepth  # midDepth  # salty |
| + Water()  + Water(string name, int s, int v, int max, int mid, bool sal)  + virtual void sq() = 0 |

|  |
| --- |
| Sea |
| # oceanClass  # borderCountry  # flowingRivers  # coastline |
| + Sea()  + Sea( string name, int s, int v, int max, int mid, bool sal, string oc, string bo, string fl, int co)  + sq() : void |

|  |
| --- |
| Ocean |
| # worldVolume |
| + Ocean()  + Ocean(string name, int s, int v,  int max, int mid, bool sal, double w)  + sq() : void |

|  |
| --- |
| Lake |
| # origin  # country  # heigtSea |
| + Lake()  + Lake(string name, int s, int v, int max,  int mid, bool sal, string ori, string co, int he)  + sq() : void |

|  |
| --- |
| River |
| # supply  # source  # mouth  # heightSource  # lenght |
| + River()  + River(string name, int s, int v, int max, int mid,  bool sal, string su, string so, string mo, int he, int le)  + sq() : void |

|  |
| --- |
| Container |
| # arrayWater  # waterCouner |
| + Container()  + maxDepthWater() : void  + sortWaterSquare() : void  + sortSalinity() : void  + addWater(Water \*newWater) |

ВИСНОВКИ: Була побудована ієрархія класів, розроблена діаграма класів. Зазначена віртуальна функція. Створений класс контейнер, який володіє динаміним списком об'єктів, має змогу додавати нові елементи в динамічний список об'єктів.