Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение  
высшего образования   
«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Отчет

по лабораторной работе № 8

**«**Паттерны поведения 1 - реализация

(State, Observer)»

по дисциплине «Объектно-ориентированное программирование»

Студент гр. ПИ-02

Чередов Р.А.

Ассистент кафедры ПМ,

Рахманин Д. С.

Барнаул 2023

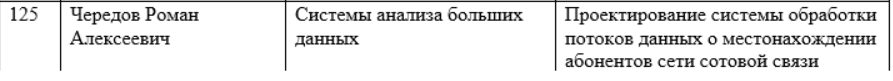
**Тема**: Паттерны поведения 1 - реализация

(State , Observer)

**Отчет**:

Логи работы программы

Архив с кодом программы



**State**

#include <iostream>

#include <typeinfo>

#include <cstdlib>

#include <ctime>

using namespace std;

class Packetinfo;

//Состояние

class State {

public:

virtual void Display(Packetinfo\* inCon) = 0; //Вывод информации

};

class Packetinfo {

private:

State\* currentState;

public:

Packetinfo(State\* inSt) {

currentState = inSt;

}

//Установить состояние

void setState(State\* inSt) {

if (currentState)

delete currentState;

currentState = inSt;

}

//Вывод состояния

void Display() {

if (currentState)

currentState->Display(this);

}

~Packetinfo() {

delete currentState;

}

};

class near : public State {

public:

void Display(Packetinfo\* inCon);

};

class direction : public State {

private:

int count; //Счетчик

int min = 0;

int max = 5;

public:

direction() {

}

virtual void Display(Packetinfo\* inCon) {

count = min + rand() % ((max + 1) - min);

if (count == 1) {

cout << "south" << endl;

};

if (count == 2) {

cout << "west" << endl;

};

if (count == 3) {

cout << "east" << endl;

};

if (count == 4) {

cout << "north" << endl;

};

if (count == 5) {

inCon->setState(new near());

};

}

};

void near::Display(Packetinfo\* inCon) {

cout << rand() << "." << rand()<<"."<< rand() << endl;

inCon->setState(new direction());

}

int main() {

State\* A = new near();

State\* B = new direction();

Packetinfo\* c = new Packetinfo(B);

for (int i = 0; i < 18; i++) {

c->Display();

}

delete A, B, c;

return 0;

}

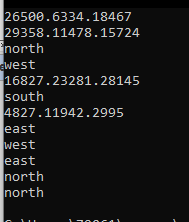


Рис. 1. Лог работы программы (State)

**Observer**

#include <iostream>

using namespace std;

class Observer {

public:

virtual void update(int v) = 0; // данные через v в update(v)

};

// конкретный наблюдатель - Scanner

class Scanner : public Observer {

private:

int oldTime, curTime;

//Анализ данных

void check() {

cout << "Time " << curTime;

if (oldTime == curTime)

cout << " hasn't changed " << endl;

else {

if (oldTime < curTime) // обработка 1

cout << " increased by " << curTime - oldTime << " milliseconds " << endl;

else // обработка 2

cout << " decreased by " << oldTime - curTime << " milliseconds " << endl;

}

}

public:

//Обновление данных

void update(int v) {

oldTime = curTime;

curTime = v;

if (oldTime >= 0) check();

}

Scanner() { curTime = -1; }

};

class Subject {

protected:

Observer\* myObserver;

public:

void attach(Observer\* a) { myObserver = a; }

void detach(Observer\* a) {

if (a == myObserver) myObserver = nullptr;

}

virtual void notify() = 0;

Subject() {}

~Subject() {}

};

// LineUp - конкретный наблюдаемый объект

class LineUp : public Subject {

private:

int t;

public:

LineUp(int start) { t = start; }

~LineUp() {}

void notify() { myObserver->update(t); }

int getState() { return t; }

void setState(int newX) {

t = newX;

notify();

}

};

int main() {

srand(time(NULL));

Observer\* obs = new Scanner();

LineUp\* LineU = new LineUp(110);

LineU->attach(obs);

//Генерация времени

for (int i = 0; i < 10; i++) {

if (i % 3)

LineU->setState(LineU->getState() + i \* rand() % 10);

else

LineU->setState(LineU->getState() - i \* rand() % 10);

}

delete obs;

delete LineU;

return 0;

}

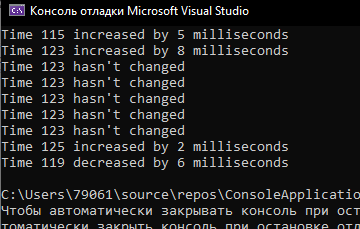


Рис. 2. Лог работы программы (Observer)