Задача 3. Створити клас TPair, який представляє пару чисел і містить методи для їх збільшення/зменшення на 1. Реалізувати класи нащадки TTime (“години.хвилини”) та TMoney (“гривні.копійки”). Згенерувати поступово випадковим чином  пар (час, гроші), де час – тривалість виконання роботи, а гроші – вартість однієї хвилини роботи працівників. Обчислити витрати на виконання кожної із робіт.

#pragma once

class Number

{

int \_value; //Поточне значення числа

int maxValue;

int minValue;

void set\_minValue(int value);

void set\_maxValue(int value);

public:

void set\_Value(int value);

int get\_Value();

bool isCorrectValue(int value);

bool increment();

Number(int value,int minValue, int maxValue);

Number(int value, int minValue);

Number(int value);

Number();

};

#include "stdafx.h"

#include "Number.h"

#include <utility>

#include <iostream>

void Number::set\_minValue(int value)

{

minValue = value;

}

void Number::set\_maxValue(int value)

{

maxValue = value;

if (minValue>maxValue)

{

std::swap(minValue, maxValue);

}

}

void Number::set\_Value(int value)

{

if (isCorrectValue(value))

this->\_value = value;

else

throw "Out of range";

}

int Number::get\_Value()

{

return \_value;

}

bool Number::isCorrectValue(int value)

{

return value>=minValue && value<=maxValue;

}

bool Number::increment()

{

\_value++;

if (\_value>maxValue)

{

\_value = minValue;

return false

}

return true;

}

Number::Number(int value, int minValue, int maxValue)

{

set\_minValue(minValue);

set\_maxValue(maxValue);

set\_Value(value);

}

Number::Number(int value, int minValue):Number(value,minValue, INT32\_MAX)

{

}

Number::Number(int value):Number(value, INT32\_MIN, INT32\_MAX)

{

}

Number::Number():Number(rand())

{

}

Number::~Number()

{

}

#pragma once

#include"Number.h"

class Pair

{

public:

Number\* number1;

Number\* number2;

Pair(int value1, int minValue1, int maxValue1,

int value2, int minValue2, int maxValue2);

bool isCorrectNumbersPair(int value1, int value2);

void increment();

};

#include "stdafx.h"

#include "Pair.h"

Pair::Pair()

{

}

Pair::Pair(int value1, int minValue1, int maxValue1, int value2, int minValue2, int maxValue2)

{

number1 = new Number(value1, minValue1, maxValue1);

number2 = new Number(value2, minValue2, maxValue2);

}

bool Pair::isCorrectNumbersPair(int value1, int value2)

{

return number1->isCorrectValue(value1) &&

number2->isCorrectValue(value2);

}

void Pair::increment()

{

if (number1->increment()==false)

{

number2->increment();

}

}

Pair::~Pair()

{

}

#pragma once

#include "Pair.h"

class Time :

protected Pair

{

public:

void set\_Minutes(int value);

void set\_Hours(int value);

int get\_Minutes();

int get\_Hours();

Time(int hours,int minutes);

Time();

};

#include "stdafx.h"

#include "Time.h"

#include<iostream>

using namespace std;

void Time::set\_Minutes(int value)

{

number1->set\_Value(value);

}

void Time::set\_Hours(int value)

{

number2->set\_Value(value);

}

int Time::get\_Minutes()

{

return number1->get\_Value();

}

int Time::get\_Hours()

{

return number2->get\_Value();

}

Time::Time(int hours, int minutes):Pair(minutes,0,59,hours,0,INT32\_MAX)

{

}

Time::Time():Time(0,0)

{

}

#pragma once

#include "Pair.h"

class Money :

protected Pair

{

public:

void set\_Coins(int value);

void set\_Grivnas(int value);

int get\_Coins();

int get\_Grivnas();

Money(int grivnas, int coins);

~Money();

};

#include "stdafx.h"

#include "Money.h"

#include<iostream>

using namespace std;

void Money::set\_Coins(int value)

{

number1->set\_Value(value);

}

void Money::set\_Grivnas(int value)

{

number2->set\_Value(value);

}

int Money::get\_Coins()

{

return number1->get\_Value();

}

int Money::get\_Grivnas()

{

return number2->get\_Value();

}

Money::Money()

{

}

Money::Money(int grivnas, int coins):Pair(coins,0,99,grivnas,0,INT32\_max)

{

}

Money::~Money()

{

}

#pragma once

#include"Money.h"

#include"Time.h"

class Task

{

public:

Time\* period;

Money\* minutesPrice;

Task();

Money getTotalPrice();

~Task();

};