Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №18.3**

Дисциплина: «Информатика»

Тема: Структуры данных

Вариант 3

Выполнила:

Студент группы ИВТ-20-2б

Солдатов Алексей Павлович

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель задачи**

Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.

Использование перегруженных операций в классах

Для достижения поставленной цели необходимо решить следующие задачи:

* Проанализировать теоретические сведения о ООП
* Проанализировать понятия классов, инструменты для их создания
* Разобрать поля и методы классов
* Понять, как пользоваться перегруженными операциями в классах
* Реализовать алгоритм поставленной задачи
* Создать отдельные файлы: .cpp для описания методов класса, .cpp для описания главной функции, .h для описания класса

Постановка задачи

Создать класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты отделяются от секунд двоеточием. Реализовать:

добавление секунд (учесть, что в минуте не может быть более 60 секунд)

вычитание секунд

Анализ задачи

1. Определить какие операции должны быть выполнены по заданию:

* Создание класса

1. class Time

* Создание Конструктора

Money::Money(long int rubs, int kops)

* Создание метода вывода информации класса
* Ввод данных

1. Для решения задачи используются переменные:

Несколько переменных целочисленного значения для ввода чисел

int rub = -1;

int kop = -1;

Поле типа double для хранения числа

double sum;

Переменные целочисленного типа для работы конструктора

long int rubs, int kops

1. Ввод данных осуществляется посредством функции

cin >> rub;

Вывод данных осуществляется посредством функции

cout << "Введи количество рублей: ";

1. Создание метода Print, который печатает данные конкретного контейнера

void Money::Print(double s)

{

cout << "Сумма = " << s << endl;

}

1. Создание контейнера

Money::Money(long int rubs, int kops)

{

double sum1 = kops;

sum1 = sum1 / 100;

sum1 = sum1 + rubs;

sum = sum1;

}

Код

Main.cpp:

#include <iostream>

#include<string>

#include<vector>

#include "Time.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

vector<Time>Time(2);

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

{

Time[i].Setmin();

Time[i].Getmin();

Time[i].Setsec();

Time[i].Getsec();

Time[i].PrintClass();

Time[i].SetDop();

Time[i].GetDop();

bool menu;

cout << "Введите 0 для прибавления секунд, введите 1 для вычитания секунд: ";

cin >> menu;

cout << endl;

switch (menu)

{

case 0: Time[i].GetSum(); break;

case 1: Time[i].GetRazn(); break;

default: break;

};

Time[i].PrintClass();

}

}

Time.cpp:

#include <iostream>

#include "Time.h"

using namespace std;

Time::Time()

{

min = 0;

sec = 0;

dop = 0;

}

Time::Time(long min, int sec)

{

this->min = min;

this->sec = sec;

this->dop = dop;

}

Time:: ~Time() {}

void Time::Setmin()

{

long val;

cout << "Введите минуты: ";

cin >> val;

while (val < 0)

{

cout << "Введите корректное число: ";

cin >> val;

}

this->min = val;

}

long Time::Getmin() { return min; }

void Time::Setsec()

{

int tmp;

cout << "Введите секунды: ";

cin >> tmp;

while (tmp > 59 || tmp < 0)

{

cout << "Введите корректное число: ";

cin >> tmp;

}

this->sec = tmp;

}

int Time::Getsec() { return sec; }

void Time::SetDop()

{

int tmp1;

cout << "Введите кол-во дополнительных секунд: ";

cin >> tmp1;

while (tmp1 > 59 || tmp1 < 0)

{

cout << "Введите корректное число: ";

cin >> tmp1;

}

this->dop = tmp1;

}

int Time::GetDop() { return dop; }

void Time::PrintClass()

{

cout << "Значение: " << this->min << ":" << this->sec << endl;

}

void Time::GetRazn()

{

if (sec >= dop)

sec -= dop;

else {

min--;

sec = 60 + sec - dop;

}

}

void Time::GetSum()

{

if (sec + dop >= 60)

{

min++;

sec = (sec + dop) - 60;

}

else

sec += dop;

}

Time& Time::operator++()

{

min++;

return \*this;

}

Time& Time::operator++(int a)

{

Time tmp(\*this);

this->min++;

return tmp;

}

Time.h:

#pragma once

#include<iostream>

using namespace std;

class Time

{

long min;

int sec;

int dop;

public:

Time();

Time(long min, int sec);

~Time();

void Setmin();

long Getmin();

Time& operator++();

Time& operator++(int a);

void Setsec();

int Getsec();

void SetDop();

int GetDop();

void PrintClass();

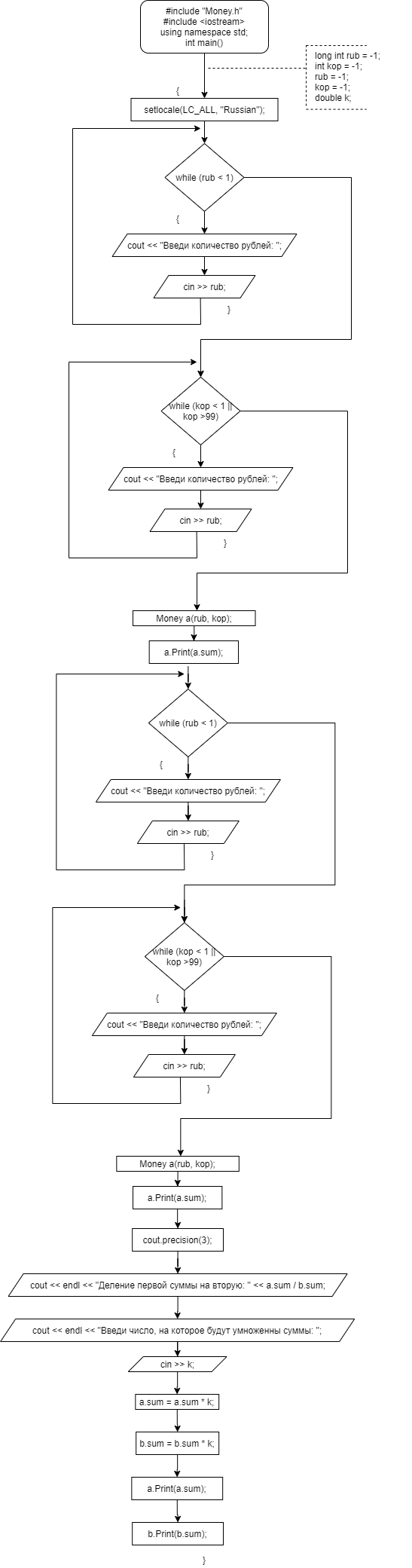
void GetRazn();

void GetSum();

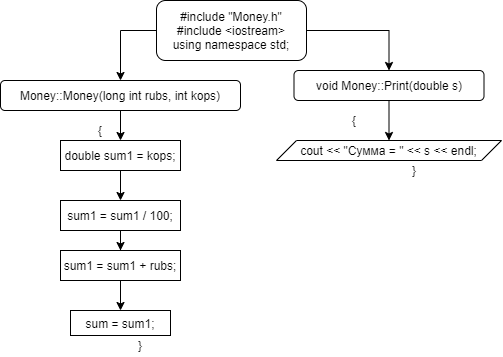
};

Блок-схема

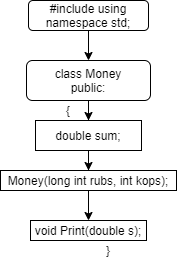
Main.cpp:



Time.cpp:



Time.h:



Работа кода

