Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ
(ТУСУР)

**Кафедра автоматизированных систем
управления (АСУ)**

Отчет по лабораторной работе № 1 по дисциплине «Объектно-ориентированное программирование»

Выполнил:
Небиев Н.И.

«_19_»_____мая_____2023 г.

Проверил: _____

_____

«_____»_____20_____
г.

Калининград 2023г

Оглавление

1. Тема работы.

Классы. Открытые и закрытые уровни доступа. Конструкторы. Инициализация данных объекта. Определение методов. Создание объекта в памяти. Стандартные потоки ввода-вывода.

2. Индивидуальное задание.

$$V = (27*64) \text{ div } 100 = 17$$

Вариант 17

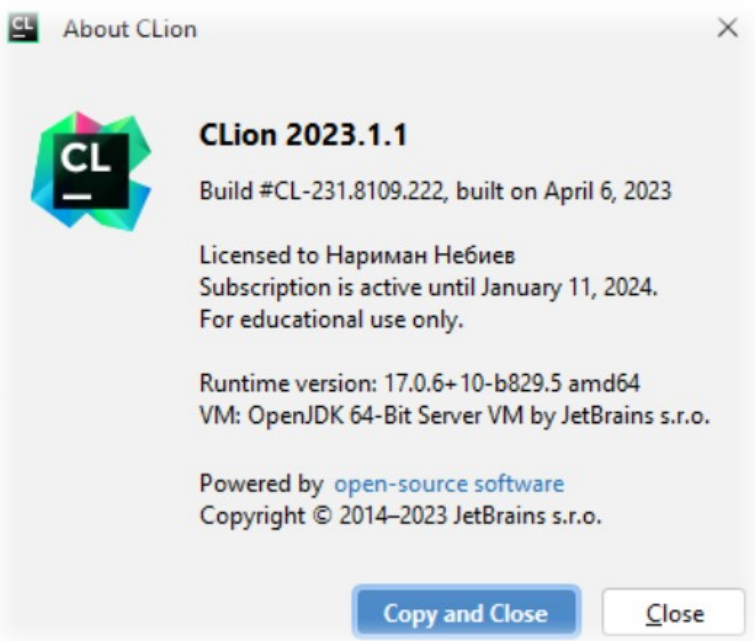
СОТОВЫЙ
марка — char
номер — int
сумма — float

3. Цель работы.

Цель данной лабораторной работы по ООП заключается в изучении классов и уровней доступа, конструкторов и методов объектов, а также создании и работы с объектами в памяти. Также в рамках работы необходимо изучить стандартные потоки ввода-вывода и научиться применять их в программировании на языке ООП.

4. Результаты работы программы.

Лабораторная работа была выполнена в интегрированной среде разработки (IDE) Clion, которая предоставляет удобный и эффективный интерфейс для написания и отладки программного кода на языке C++. Использование данной IDE позволило мне значительно ускорить процесс разработки, благодаря быстрому доступу к различным инструментам и функциям, таким как автоматическое завершение кода, подсветка синтаксиса, отладчик и многие другие.



При запуске программы создаем объект p1 и выводим его поля на экран:

```
Brand: Samsung
Number: 123456
Sum: 1000
```

Далее создаем объект p2 и выводим его поля на экран:

```
Brand: Apple
Number: 654321
Sum: 2000
```

Затем меняем значения полей с помощью методов SetBrand, SetNumber, SetSum и выводим новые значения на экран с помощью методов GetBrand, GetNumber, GetSum:

```
New brand: Google
New sum: 3000
```

После программа потребует указать brend, number, sum. При этом программа проверит вводимые значение number и sum:

```
Enter brand: ii
Enter number: dasdas
Enter number: 10
Enter sum: adasdas
Enter sum: 10
Brand: ii
Number: 10
Sum: 10
```

5. Выводы.

В ходе выполнения данной лабораторной работы мы изучили классы и уровни доступа, конструкторы и методы объектов, а также создание и работу с объектами в памяти на языке ООП. Мы также освоили применение стандартных потоков ввода-вывода и научились использовать их в программировании.

В результате выполнения работы мы достигли поставленных целей и ответили на все поставленные в начале работы вопросы. Мы приобрели новые знания и навыки, которые могут быть использованы в будущих проектах, связанных с программированием на языке ООП.

Таким образом, выполнение данной лабораторной работы позволило нам более глубоко изучить основы объектно-ориентированного программирования и научиться применять их на практике.

Приложение А

main.h:

```
#include <iostream>
#include <cstring>

#pragma once
#ifndef PERSON_H // Директива условной компиляции, чтобы header-файл был
включен только один раз
#define PERSON_H

#include <iostream>
#include <cstring>
#include <limits>

class Person {
private:
    char* brand;
    int number;
    float sum;
public:
    // Конструкторы и деструктор
    Person(const char* b, int n, float s);
    Person();
    ~Person();
    // Методы вывода и ввода объекта
    void Print() const;
    void Input();
    // Методы установки и получения значений полей объекта
    void SetBrand(const char* b);
    void SetNumber(int n);
    void SetSum(float s);
    const char* GetBrand() const;
    int GetNumber() const;
    float GetSum() const;
};

#endif


main.cpp:
#include "main.h"


// Функция для проверки ввода числа
template<typename T>
bool CheckInput(T& var) {
    std::cin >> var;
    if (std::cin.fail()) { // если ввод не удался
        std::cin.clear(); // очищаем флаг ошибки
        std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\
n'); // удаляем лишние символы из буфера
        return false;
    }
    std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n'); //
удаляем лишние символы из буфера
    return true;
}
```

```
// Конструктор, выделяющий память под brand и копирующий строку b в неё
Person::Person(const char* b, int n, float s) {
    brand = new char[strlen(b) + 1];
    strcpy(brand, b);
    number = n;
    sum = s;
}

// Конструктор по умолчанию, создающий объект со значениями переменных по
умолчанию
Person::Person() {
    brand = new char[1];
    strcpy(brand, "");
    number = 0;
    sum = 0.0;
}

// Деструктор, освобождающий память, выделенную под brand
Person::~Person() {
    delete[] brand;
}

// Метод вывода полей объекта на экран
void Person::Print() const {
    std::cout << "Brand: " << brand << std::endl;
    std::cout << "Number: " << number << std::endl;
    std::cout << "Sum: " << sum << std::endl;
}

// Метод ввода полей объекта с клавиатуры
void Person::Input() {
    char buf[100];
    std::cout << "Enter brand: ";
    std::cin.getline(buf, 100);
    // Освобождаем старую память, выделенную под brand
    delete[] brand;
    // Выделяем новую память под brand и копируем строку buf в неё
    brand = new char[strlen(buf) + 1];
    strcpy(brand, buf);

    // Проверка ввода числа
    do {
        std::cout << "Enter number: ";
    } while (!CheckInput(number));

    // Проверка ввода числа с плавающей точкой
    do {
        std::cout << "Enter sum: ";
    } while (!CheckInput(sum));
}

// Метод установки значения поля brand
void Person::SetBrand(const char* b) {
    // Освобождаем старую память, выделенную под brand
    delete[] brand;
    // Выделяем новую память под brand и копируем строку b в неё
    brand = new char[strlen(b) + 1];
    strcpy(brand, b);
}

// Метод установки значения поля number
void Person::SetNumber(int n) {
```

```
    number = n;
}

// Метод установки значения поля sum
void Person::SetSum(float s) {
    sum = s;
}

// Метод получения значения поля brand
const char* Person::GetBrand() const {
    return brand;
}

// Метод получения значения поля number
int Person::GetNumber() const {
    return number;
}

// Метод получения значения поля sum
float Person::GetSum() const {
    return sum;
}

int main() {
    // Создаем объект p1 и выводим его поля на экран
    Person p1("Samsung", 123456, 1000.0);
    p1.Print();


    // Создаем объект p2 и выводим его поля на экран.
    // Затем меняем значения полей с помощью методов SetBrand и SetSum,
    // и выводим новые значения на экран с помощью методов GetBrand и GetSum.
    Person p2("Apple", 654321, 2000.0);
    p2.Print();
    p2.SetBrand("Google");
    p2.SetSum(3000.0);
    std::cout << "New brand: " << p2.GetBrand() << std::endl;
    std::cout << "New sum: " << p2.GetSum() << std::endl;


    // Создаем объект p3, считываем его поля с помощью метода Input,
    // и выводим их на экран с помощью метода Print.
    Person p3;
    p3.Input();
    p3.Print();


    return 0;
}
```