Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

ОТЧЕТ

к лабораторной работе №1

**«КЛАССЫ И ОБЪЕКТЫ. ОРГАНИЗАЦИЯ**

**ВВОДА/ВЫВОДА. ДИНАМИЧЕСКОЕ ВЫДЕЛЕНИЕ**

**ПАМЯТИ В ЯЗЫКЕ С++»**

|  |  |
| --- | --- |
|  | |
|  |
|  |
|  |
|  |  | | |
|  | | |  | |  |
|  | | |  | |  |
| Студент гр. 324402 | | |  | | Цевелюк А. И. |
| Проверил | | |  | | Булыга Д. И. |

Минск 2024

**1 Цель работы**

Цель работы:изучить структуру класса, атрибуты доступа к  
компонентам класса; рассмотреть принцип работы конструкторов (с  
параметрами, без параметров, с параметрами по умолчанию, конструктора  
копирования), деструкторов при работе с объектом, статические и  
константные данные и методы класса.

**2 Теоретические сведения по лабораторной работе**

Лабораторная работа направлена на изучение структуры класса, атрибутов доступа к его компонентам, конструкторов (с параметрами, без параметров, с параметрами по умолчанию, конструктора копирования) и деструкторов. Также важным аспектом является работа с динамическим выделением памяти. Ключевыми понятиями являются классы, объекты, методы, конструкторы, деструкторы, атрибуты доступа, динамическое выделение памяти, ввод/вывод данных.

**3 Формулировка индивидуального задания**

Создать класс Bot (бот). В классе должны быть следующие поля: имя бота, версия бота, сфера применения. Создать методы формирования и отправки ответа на полученный запрос, сохранения всех отправленных ответов. В класс добавить необходимый набор полей и методов (минимум два поля и два метода) на свое усмотрение.

**4 Код решения индивидуального задания**

#include <iostream>

#include <string>

#include <fstream>

#include <ctime>

#include <iomanip>

class bot

{

private:

std::string name\_, version\_, sphere\_, author\_;

bool is\_logging\_;

public:

bot();

bot(const std::string& name, const std::string& version, const std::string& sphere, const std::string& author, bool is\_logging);

bot(const bot& obj);

void log(const std::string& message, int format) const;

void ping() const;

void print\_info() const;

void echo() const;

int choose\_option() const;

void say\_goodbye() const;

};

bot::bot(const std::string& name, const std::string& version, const std::string& sphere, const std::string& author, const bool is\_logging)

{

name\_ = name;

version\_ = version;

sphere\_ = sphere;

author\_ = author;

is\_logging\_ = is\_logging;

}

bot::bot()

{

name\_ = "NoName Bot";

version\_ = "0.0";

sphere\_ = "simple bot";

author\_ = "NoName author";

is\_logging\_ = true;

}

bot::bot(const bot& obj)

{

name\_ = obj.name\_;

version\_ = obj.version\_;

sphere\_ = obj.sphere\_;

author\_ = obj.author\_;

is\_logging\_ = obj.is\_logging\_;

}

void bot::log(const std::string& message, const int format) const

{

if (is\_logging\_)

{

std::string format\_type;

std::ofstream out;

out.open("lab1\_logs.txt", std::ios::app);

struct tm time;

const time\_t now = std::time(nullptr);

localtime\_s(&time, &now);

switch (format)

{

case 0:

format\_type = name\_;

break;

case 1:

format\_type = "USER";

break;

}

out << "[" << std::put\_time(&time, "%d-%m-%Y %H-%M-%S") << "] " << format\_type << ": " << message << '\n';

out.close();

}

}

void bot::ping() const

{

const std::string message = "Pong! We're on local, so your ping is 0ms.";

std::cout << message << '\n';

log(message, 0);

}

void bot::print\_info() const

{

std::string logging;

if (is\_logging\_)

logging = "true";

else

logging = "false";

const std::string message = "Hello! My name is " + name\_ + ", version " + version\_ + ". My author is " + author\_ + ". My sphere is " + sphere\_ + ". My logging parametr is " + logging + ".";

std::cout << message << '\n';

log(message, 0);

}

void bot::echo() const

{

std::string temp;

const std::string message = "Enter your message: ";

std::cout << message;

log(message, 0);

std::cin.ignore();

std::getline(std::cin, temp);

log(temp, 1);

std::cout << temp << '\n';

log(temp, 0);

}

int bot::choose\_option() const

{

int choice = 0;

const std::string message = "Choose next options: 1 - print info 'bout me, 2 - ping me, 3 - echo bot, any else - exit: ";

std::cout << message;

std::cin >> choice;

log(std::to\_string(choice), 1);

switch (choice)

{

case 1:

return 1;

case 2:

return 2;

case 3:

return 3;

default:

return -1;

}

}

void bot::say\_goodbye() const

{

const std::string message = "Goodbye, see you soon!";

std::cout << message << '\n';

log(message, 0);

}

int main()

{

srand(static\_cast<int>(time(nullptr)));

const auto bots = new bot[5]

{

bot("Bread Bot", "0.1", "simple answer bot", "Aliaksiei Tsevialiuk", true),

bot("Lemon Bot", "1.0", "test bot", "Alex\_Tsev", false),

bot("Skuf Bot", "0.5a", "answer bot", "Aliaksiei", true),

bot("Open Source Bot", "24w6a", "answer", "Tsevialiuk", false),

bot()

};

const bot bot = bots[rand() % 5];

int global\_choice = 0;

bot.print\_info();

do

{

global\_choice = bot.choose\_option();

switch (global\_choice)

{

case 1:

bot.print\_info();

break;

case 2:

bot.ping();

break;

case 3:

bot.echo();

break;

}

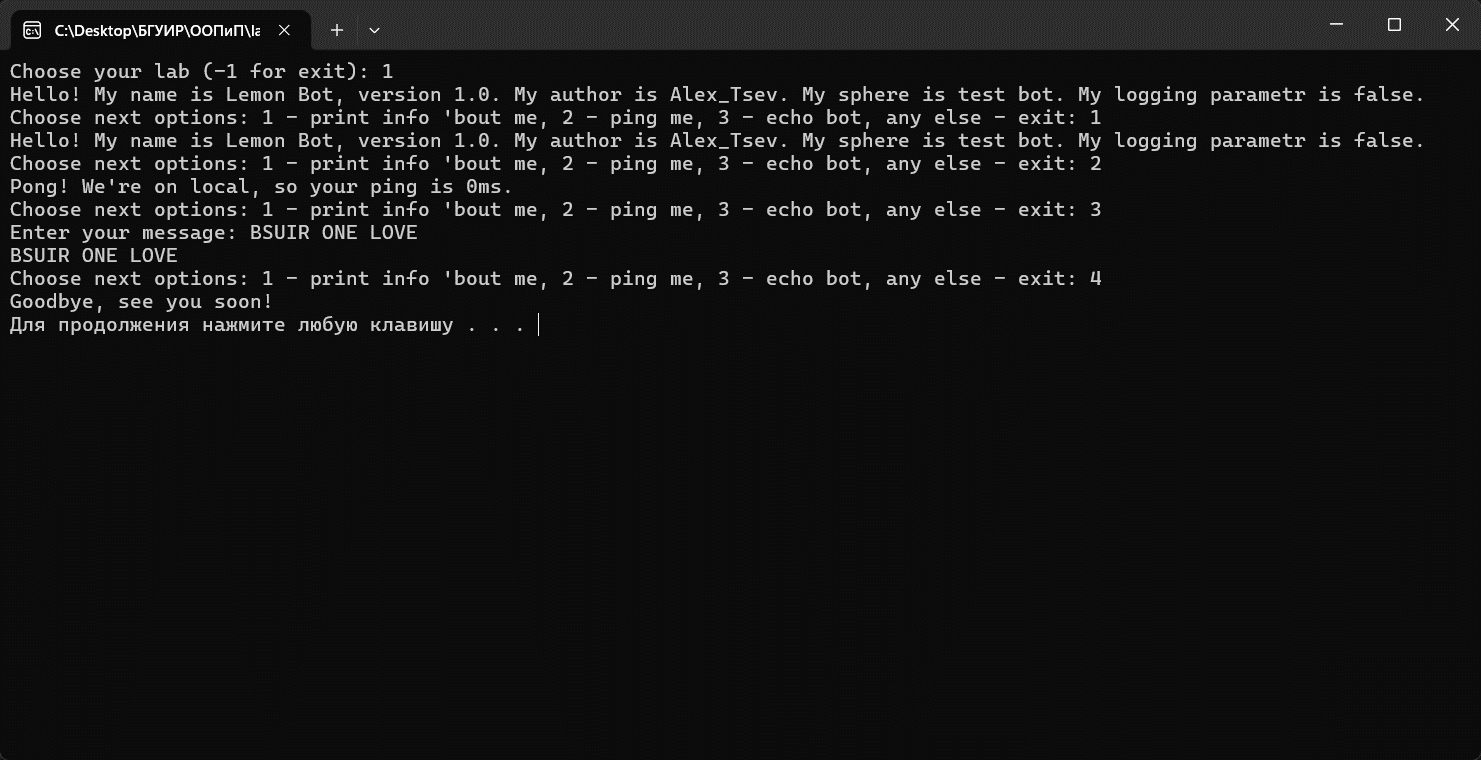
} while (global\_choice != -1);

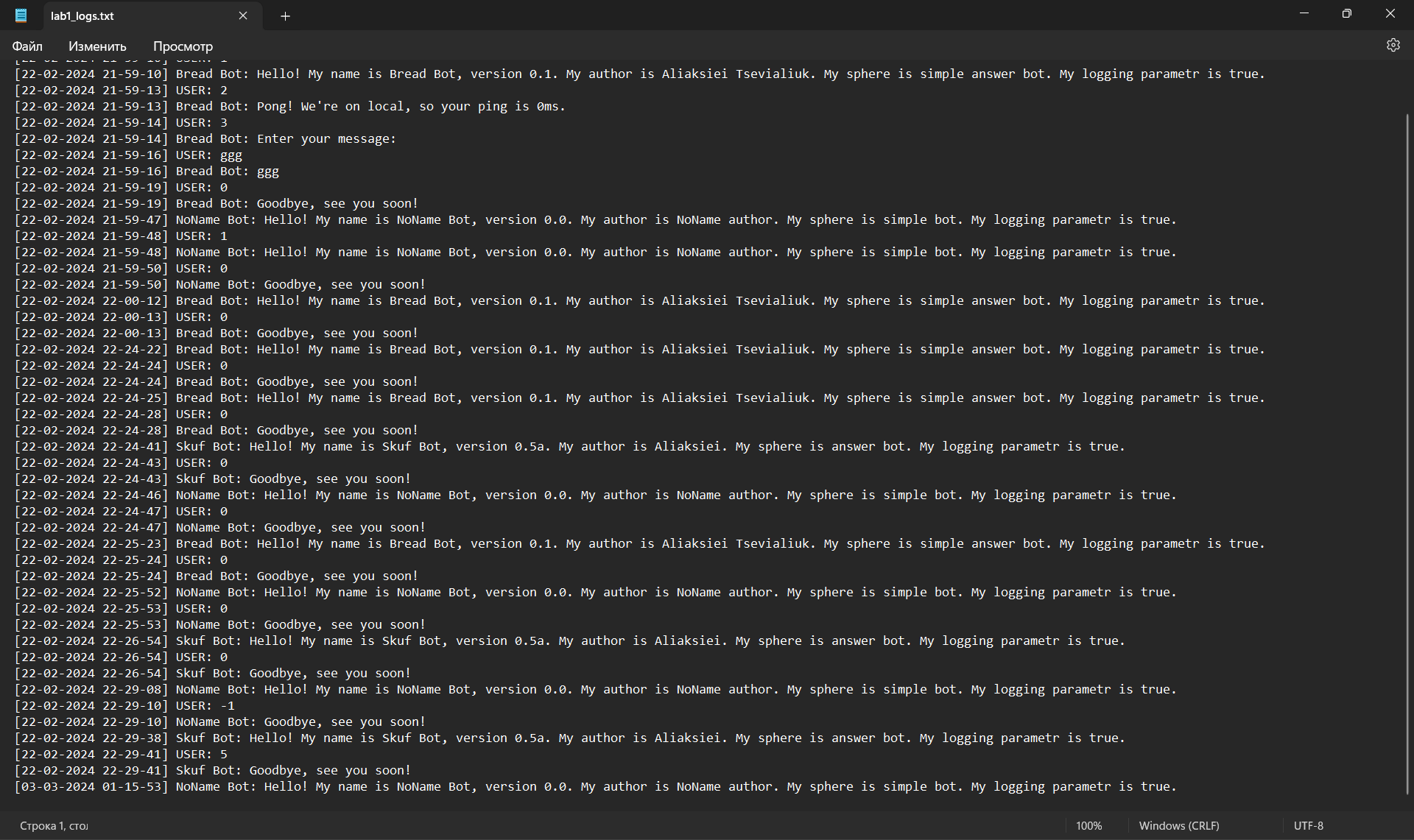
bot.say\_goodbye();

return 0;

}

**5 Скриншоты выполнения индивидуального задания**





**6 Ответы на контрольные вопросы**

1. В чем разница между пользовательскими типами данных struct, class, union?

Разница между пользовательскими типами данных struct, class, union заключается в области видимости по умолчанию: у struct и union доступ по умолчанию public, в то время как у class - private. Кроме того, struct и union не являются частью парадигмы ООП.

1. Что такое указатель this? Приведите пример использования этого указателя.

Указатель this — это указатель на текущий объект. Пример использования: в методах класса this используется для доступа к членам объекта.

1. Когда вызывается конструктор объекта?

Конструктор объекта вызывается при создании нового объекта данного класса.

1. Когда вызывается деструктор объекта?

Деструктор объекта вызывается при уничтожении объекта: когда он выходит за область видимости или явно удаляется.

1. Какова основная форма конструктора копирования и перечислите ситуации, в которых он вызывается?

Основная форма конструктора копирования: ClassName(const ClassName& obj). Он вызывается при инициализации нового объекта существующим объектом того же класса, при передаче объекта по значению и возвращении объекта из функции по значению.

1. Можно ли конструктор и деструктор перегрузить?

Конструкторы могут быть перегружены для различных наборов параметров, но деструктор не может быть перегружен, так как он не имеет параметров.

1. Приведите пример использования константных и статических данных и методов.

Константные данные и методы могут быть использованы для представления постоянных значений или операций, которые не должны изменяться в течение работы программы. Например, константный метод может возвращать значение PI. Статические данные и методы принадлежат классу в целом, а не конкретному объекту. Они могут использоваться для общих операций или хранения общих данных между всеми объектами данного класса.

1. Приведите пример использования локальных и глобальных статических переменных.

Локальная статическая переменная сохраняет свое значение между вызовами функции. Например, счетчик вызовов функции. Глобальная статическая переменная доступна в пределах файла, в котором она объявлена, и сохраняет свое значение на протяжении всей программы. Например, глобальная статическая переменная для хранения общих данных между несколькими функциями.

1. В чем различие передачи аргумента по ссылке и по значению?

При передаче аргумента по значению создается копия значения аргумента, что может быть затратно по памяти и времени, особенно для больших объектов. При передаче аргумента по ссылке передается сам адрес объекта, что позволяет изменять его значение в вызывающей функции без создания копии.

1. Создается ли конструктор по умолчанию, если в классе создан один конструктор с параметрами?

Если в классе создан хотя бы один конструктор с параметрами, то компилятор не создает конструктор по умолчанию. В этом случае необходимо явно определить конструктор по умолчанию, если он требуется.

**7 Выводы по лабораторной работе**

Вывод: я изучил структуру класса, атрибуты доступа к компонентам класса, рассмотрел принцип работы конструкторов (с параметрами, без параметров, с параметрами по умолчанию, конструктора копирования), деструкторов при работе с объектом, статических и константных данных и методов класса.