Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новосибирский государственный технический университет»

Кафедра Автоматизированных систем управления

**Практическая работа**

по дисциплине «Программирование»

Тема: Классы, исключения

|  |  |
| --- | --- |
| Выполнили:  Кривощапов Сергей,  Пирожников Владислав,  Писарев Радомир  Группа АВТ-414  Факультет АВТФ.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «13» мая 2025 г. | Проверил:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Ф. И. О.)  Балл: \_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |

Новосибирск 2025

[Ответы на вопросы по шагам 1 – 3. 3](#_Toc1)

[1) Определить имена класса и его полей. 3](#_Toc2)

[2) Определить тип данных для полей класса. 3](#_Toc3)

[3) Определить множество допустимых значений для полей класса. 3](#_Toc4)

[Определение класса с комментариями о назначении полей и методов (шаги 5 – 10). 4](#_Toc5)

[Тексты разработанных методов (шаги 5 – 10). Комментарии об особенностях их работы (алгоритм). 5](#_Toc6)

[Метод Init() 5](#_Toc7)

[Конструктор с параметрами 5](#_Toc8)

[Конструктор копирования 5](#_Toc9)

[Метод print\_router\_data 6](#_Toc10)

[Метод change\_vendor 6](#_Toc11)

[Метод change\_port\_count 6](#_Toc12)

[Метод change\_has\_5g 6](#_Toc13)

[Функция create\_router 6](#_Toc14)

[Функция field\_changing 7](#_Toc15)

[Текст тестирующей функции (шаг 11) с комментариями. 9](#_Toc16)

[Скриншот результата запуска тестовой функции и выводы о полученных результатах. 10](#_Toc17)

[Скриншот 10](#_Toc18)

[Выводы о полученных результатах 10](#_Toc19)

[Приложение. Исходный код программы. 11](#_Toc20)

[wifi.h 11](#_Toc21)

[wifi.cpp 11](#_Toc22)

[main.cpp 13](#_Toc23)

# Ответы на вопросы по шагам 1 – 3.

## 1) Определить имена класса и его полей.

Класс Wifi

vendor - производитель

port\_count - количество портов

has\_5g - наличие поддержки 5G

## 2) Определить тип данных для полей класса.

vendor:

Тип: std::string (строка)

port\_count:

Тип: unsigned int (неотрицательное целое число)

has\_5g:

Тип: bool (логический)

## 3) Определить множество допустимых значений для полей класса.

vendor: не может быть пустой

port\_count: 1 - 16

has\_5g: false/true

# Определение класса с комментариями о назначении полей и методов (шаги 5 – 10).

|  |
| --- |
| class Wifi { private:  std::string vendor; // производитель  unsigned int port\_count; // количество портов  bool has\_5g; // наличие поддержки 5G  // метод для инициализации полей класса   void init(const std::string& vendor, unsigned int port\_count, bool has\_5g); public:  // конструктор по умолчанию  Wifi(const std::string& vendor = "Undefined", unsigned int port\_count = 1, bool has\_5g = false);  // конструктор копирования  Wifi(const Wifi& router\_to\_copy);  // деструктор  ~Wifi() = default;  // методы доступа  std::string get\_vendor() const {return vendor;};  unsigned int get\_port\_count() const {return port\_count;};  bool get\_has\_5g() const {return has\_5g;};  // вывод сведений о роутере  void print\_router\_data() const;  void change\_vendor(const std::string& new\_vendor); // метод для изменение имени производителя  void change\_port\_count(unsigned int new\_port\_count); // метод для изменение количества портов  void change\_has\_5g(bool new\_has\_5g); // метод для изменение метки поддержки 5G }; Wifi\* create\_router(); // функция для инициализации экземпляра класса в динамической памяти void field\_changing(Wifi& router); // функция для изменения поля экземпляра класса |

# Тексты разработанных методов (шаги 5 – 10). Комментарии об особенностях их работы (алгоритм).

## Метод Init()

|  |
| --- |
| void Wifi::init(const std::string& vendor, unsigned int port\_count, bool has\_5g) {  if (vendor.empty()) throw std::invalid\_argument("Vendor cannot be empty");  if (port\_count < 1 || port\_count > 16) throw std::out\_of\_range("Port count cannot be less than 1 and more than 16");  this->vendor = vendor;  this->port\_count = port\_count;  this->has\_5g = has\_5g; } |

Проверить что vendor не пустой (или исключение)

Проверить port\_count в диапазоне 1-16 (или исключение)

Установить значения полей объекта

## Конструктор с параметрами

|  |
| --- |
| Wifi::Wifi(const std::string& vendor, unsigned int port\_count, bool has\_5g) {  init(vendor, port\_count, has\_5g); } |

Вызвать метод init() с переданными параметрами

## Конструктор копирования

|  |
| --- |
| Wifi::Wifi(const Wifi& router\_to\_copy) {  init(router\_to\_copy.vendor, router\_to\_copy.port\_count, router\_to\_copy.has\_5g); } |

Вызвать init() со значениями из объекта-оригинала

## Метод print\_router\_data

|  |
| --- |
| void Wifi::print\_router\_data() const{  std::cout << "Vendor name: " << vendor << std::endl;  std::cout << "Amount of ports: " << port\_count << std::endl;  std::cout << "Has 5G mark? " << (has\_5g ? "Yes" : "No" ) << std::endl; } |

Вывести название производителя

Вывести количество портов

Вывести метку 5G (преобразовав в yes/no)

## Метод change\_vendor

|  |
| --- |
| void Wifi::change\_vendor(const std::string& new\_vendor) {  init(new\_vendor, port\_count, has\_5g); } |

Вызвать init() с новым значением для vendor

## Метод change\_port\_count

|  |
| --- |
| void Wifi::change\_port\_count(unsigned int new\_port\_count) {  init(vendor, new\_port\_count, has\_5g); } |

Вызвать init() с новым значением для port\_count

## Метод change\_has\_5g

|  |
| --- |
| void Wifi::change\_has\_5g(bool new\_has\_5g) {  init(vendor, port\_count, new\_has\_5g); } |

Вызвать init() с новым значением для has\_5g

## Функция create\_router

|  |
| --- |
| Wifi\* create\_router() {  std::string vendor;  unsigned int port\_count;  char has\_5g;  try {  std::cout << "Enter vendor name: ";  std::getline(std::cin, vendor);  }catch (std::invalid\_argument& e) {return nullptr;}  std::cout << "Enter amount of ports (1-16): ";  std::cin >> port\_count;  std::cin.ignore();  std::cout << "Has 5G mark? (y/n): ";  std::cin >> has\_5g;  std::cin.ignore();  try {  if (has\_5g == 'y' || has\_5g == 'Y') {  Wifi\* router = new Wifi(vendor, port\_count, true);  return router;  }  else if (has\_5g == 'n' || has\_5g == 'N') {  Wifi\* router = new Wifi(vendor, port\_count, false);  return router;  }  throw std::invalid\_argument("5g mark is able to be only y/n");  } catch (const std::invalid\_argument& exp) {  std::cout << "Invalid argument: " << exp.what() << std::endl;  return nullptr;  } } |

Запросить у пользователя vendor

Запросить port\_count

Запросить has\_5g

Если введенные значения корректны, то создать новый объект в динамической памяти

И вернуть указатель или

Вернуть nullptr при ошибке

## Функция field\_changing

|  |
| --- |
| void field\_changing(Wifi& router) {  std::cout << "Which field do you want to change?" << std::endl<< "1 = vendor" << std::endl << "2 = port count" << std::endl <<  "3= 5G mark" << std::endl;  int field\_to\_change;  std::cin >> field\_to\_change;  std::cin.ignore();  switch (field\_to\_change) {  case 1: {  std::string new\_vendor;  std::cout << "Enter new vendor name: ";  std::getline(std::cin, new\_vendor);  router.change\_vendor(new\_vendor);  break;  }  case 2: {  unsigned int new\_port;  std::cout << "Enter new port count (1-16): ";  std::cin >> new\_port;  std::cin.ignore();  router.change\_port\_count(new\_port);  break;  }  case 3: {  char has\_5g;  std::cout << "Has 5G mark? (y/n): ";  std::cin >> has\_5g;  std::cin.ignore();  if (has\_5g == 'y' || has\_5g == 'Y') {  router.change\_has\_5g(true);  }  else if (has\_5g == 'n' || has\_5g == 'N') {  router.change\_has\_5g(false);  }  else {  throw std::invalid\_argument("5G mark must be y or n");  }  break;  }  default:  throw std::out\_of\_range("Choice must be 1, 2 or 3");  }  std::cout << "Router after changes: " << std::endl;  router.print\_router\_data(); } |

Вывести меню выбора поля

Получить на ввод выбор пользователя (1-3)

Запросить новое значение

Вызвать соответствующий метод изменения

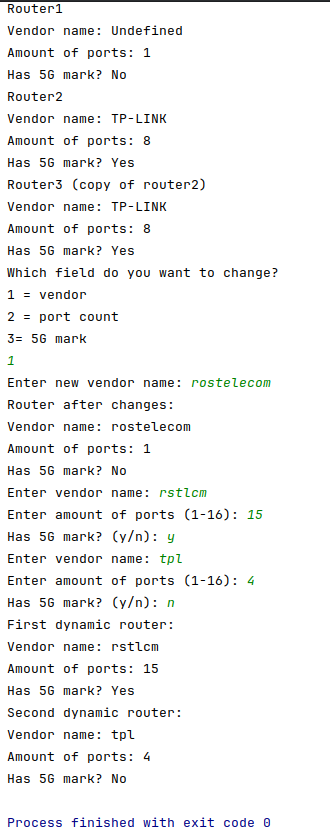
Вывести обновленные данные роутера

# Текст тестирующей функции (шаг 11) с комментариями.

|  |
| --- |
| int main() {  try {  // создание экземпляров класса на стеке  Wifi router1;  Wifi router2("TP-LINK", 8, true);  // вывод сведений об объектах  std::cout << "Router1" << std::endl;  router1.print\_router\_data();  std::cout << "Router2" << std::endl;  router2.print\_router\_data();  Wifi router3(router2); // инициализация экземпляра как копии другого  std::cout << "Router3 (copy of router2)" << std::endl;  router3.print\_router\_data();  field\_changing(router1); // вызов функции изменения поля  // вызов функции инициализации в динамической памяти  Wifi\* dynamic\_router1 = create\_router();  Wifi\* dynamic\_router2 = create\_router();  // проверка на успешность инициализации, вывод сведений о инициализированном экземпляре  if (dynamic\_router1) {  std::cout << "First dynamic router: " << std::endl;  dynamic\_router1->print\_router\_data();  } else {  std::cout << "First dynamic router creation failed" << std::endl;  }  if (dynamic\_router2) {  std::cout << "Second dynamic router: " << std::endl;  dynamic\_router2->print\_router\_data();  } else {  std::cout << "Second dynamic router creation failed" << std::endl;  }  // освобождение динамической памяти  delete dynamic\_router1;  delete dynamic\_router2;  }  // обработка исключений  catch (std::exception& exp) {  std::cout << exp.what() << std::endl;  return 1;  }  return 0; } |

# Скриншот результата запуска тестовой функции и выводы о полученных результатах.

## Скриншот



## Выводы о полученных результатах

Методы функционируют корректно, проверка значений выполняется, обработка исключений реализована корректно.

# Приложение. Исходный код программы.

## wifi.h

|  |
| --- |
| #pragma once #include <iostream> #include <string> #include <stdexcept> class Wifi { private:  std::string vendor; // производитель  unsigned int port\_count; // количество портов  bool has\_5g; // наличие поддержки 5G  // метод для инициализации полей класса  void init(const std::string& vendor, unsigned int port\_count, bool has\_5g); public:  // конструктор по умолчанию  Wifi(const std::string& vendor = "Undefined", unsigned int port\_count = 1, bool has\_5g = false);  // конструктор копирования  Wifi(const Wifi& router\_to\_copy);  // деструктор  ~Wifi() = default;  // методы доступа  std::string get\_vendor() const {return vendor;};  unsigned int get\_port\_count() const {return port\_count;};  bool get\_has\_5g() const {return has\_5g;};  // вывод сведений о роутере  void print\_router\_data() const;  void change\_vendor(const std::string& new\_vendor); // метод для изменение имени производителя  void change\_port\_count(unsigned int new\_port\_count); // метод для изменение количества портов  void change\_has\_5g(bool new\_has\_5g); // метод для изменение метки поддержки 5G }; Wifi\* create\_router(); // функция для инициализации экземпляра класса в динамической памяти void field\_changing(Wifi& router); // функция для изменения поля экземпляра класса |

## wifi.cpp

|  |
| --- |
| #include "wifi.h" void Wifi::init(const std::string& vendor, unsigned int port\_count, bool has\_5g) {  if (vendor.empty()) throw std::invalid\_argument("Vendor cannot be empty");  if (port\_count < 1 || port\_count > 16) throw std::out\_of\_range("Port count cannot be less than 1 and more than 16");  this->vendor = vendor;  this->port\_count = port\_count;  this->has\_5g = has\_5g; } Wifi::Wifi(const std::string& vendor, unsigned int port\_count, bool has\_5g) {  init(vendor, port\_count, has\_5g); } Wifi::Wifi(const Wifi& router\_to\_copy) {  init(router\_to\_copy.vendor, router\_to\_copy.port\_count, router\_to\_copy.has\_5g); } void Wifi::print\_router\_data() const{  std::cout << "Vendor name: " << vendor << std::endl;  std::cout << "Amount of ports: " << port\_count << std::endl;  std::cout << "Has 5G mark? " << (has\_5g ? "Yes" : "No" ) << std::endl; } void Wifi::change\_vendor(const std::string& new\_vendor) {  init(new\_vendor, port\_count, has\_5g); } void Wifi::change\_port\_count(unsigned int new\_port\_count) {  init(vendor, new\_port\_count, has\_5g); } void Wifi::change\_has\_5g(bool new\_has\_5g) {  init(vendor, port\_count, new\_has\_5g); } Wifi\* create\_router() {  std::string vendor;  unsigned int port\_count;  char has\_5g;  try {  std::cout << "Enter vendor name: ";  std::getline(std::cin, vendor);  }catch (std::invalid\_argument& e) {return nullptr;}  std::cout << "Enter amount of ports (1-16): ";  std::cin >> port\_count;  std::cin.ignore();  std::cout << "Has 5G mark? (y/n): ";  std::cin >> has\_5g;  std::cin.ignore();  try {  if (has\_5g == 'y' || has\_5g == 'Y') {  Wifi\* router = new Wifi(vendor, port\_count, true);  return router;  }  else if (has\_5g == 'n' || has\_5g == 'N') {  Wifi\* router = new Wifi(vendor, port\_count, false);  return router;  }  throw std::invalid\_argument("5g mark is able to be only y/n");  } catch (const std::invalid\_argument& exp) {  std::cout << "Invalid argument: " << exp.what() << std::endl;  return nullptr;  } } void field\_changing(Wifi& router) {  std::cout << "Which field do you want to change?" << std::endl<< "1 = vendor" << std::endl << "2 = port count" << std::endl <<  "3= 5G mark" << std::endl;  int field\_to\_change;  std::cin >> field\_to\_change;  std::cin.ignore();  switch (field\_to\_change) {  case 1: {  std::string new\_vendor;  std::cout << "Enter new vendor name: ";  std::getline(std::cin, new\_vendor);  router.change\_vendor(new\_vendor);  break;  }  case 2: {  unsigned int new\_port;  std::cout << "Enter new port count (1-16): ";  std::cin >> new\_port;  std::cin.ignore();  router.change\_port\_count(new\_port);  break;  }  case 3: {  char has\_5g;  std::cout << "Has 5G mark? (y/n): ";  std::cin >> has\_5g;  std::cin.ignore();  if (has\_5g == 'y' || has\_5g == 'Y') {  router.change\_has\_5g(true);  }  else if (has\_5g == 'n' || has\_5g == 'N') {  router.change\_has\_5g(false);  }  else {  throw std::invalid\_argument("5G mark must be y or n");  }  break;  }  default:  throw std::out\_of\_range("Choice must be 1, 2 or 3");  }  std::cout << "Router after changes: " << std::endl;  router.print\_router\_data(); } |

## main.cpp

|  |
| --- |
| #include "wifi.h" int main() {  try {  // создание экземпляров класса на стеке  Wifi router1;  Wifi router2("TP-LINK", 8, true);  // вывод сведений об объектах  std::cout << "Router1" << std::endl;  router1.print\_router\_data();  std::cout << "Router2" << std::endl;  router2.print\_router\_data();  Wifi router3(router2); // инициализация экземпляра как копии другого  std::cout << "Router3 (copy of router2)" << std::endl;  router3.print\_router\_data();  field\_changing(router1); // вызов функции изменения поля  // вызов функции инициализации в динамической памяти  Wifi\* dynamic\_router1 = create\_router();  Wifi\* dynamic\_router2 = create\_router();  // проверка на успешность инициализации, вывод сведений о инициализированном экземпляре  if (dynamic\_router1) {  std::cout << "First dynamic router: " << std::endl;  dynamic\_router1->print\_router\_data();  } else {  std::cout << "First dynamic router creation failed" << std::endl;  }  if (dynamic\_router2) {  std::cout << "Second dynamic router: " << std::endl;  dynamic\_router2->print\_router\_data();  } else {  std::cout << "Second dynamic router creation failed" << std::endl;  }  // освобождение динамической памяти  delete dynamic\_router1;  delete dynamic\_router2;  }  // обработка исключений  catch (std::exception& exp) {  std::cout << exp.what() << std::endl;  return 1;  }  return 0; } |