|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**КУРСОВОЙ ПРОЕКТ (РАБОТА)**

по дисциплине: Объектно-ориентированное программирование

по профилю:Прикладная информатика

направления профессиональной подготовки:Прикладная информатика, бакалавриат

Тема: «Программа для управления деятельностью агентства по подбору персонала.»

Студент (ф.и.о. полностью): Ватулин Дмитрий Сергеевич

Группа: ИНБО-04-18

Работа представлена к защите\_\_\_\_\_\_\_\_\_\_\_(дата)\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись и ф.и.о. студента)

Руководитель: Ассистент кафедры ИиППО Хлебникова В.Л

Работа допущена к защите\_\_\_\_\_\_\_\_\_\_\_(дата)\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись ф.и.о. руководителя)

Оценка по итогам защиты: \_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /

(подписи, дата, ф.и.о., должность, звание, уч. степень двух преподавателей, принявших защиту)

М. МИРЭА. 2019г.

УДК 004.045

Ватулин Д.С.Разработка курсовой работы **/ Курсовая работа** по дисциплине «Объектно-ориентированное программирование» профиля «Информатизация организаций» направления профессиональной подготовки бакалавриата 09.03.03. «Прикладная информатика» ( II семестр) / руководитель ассистент кафедра ИиППО В. Л. Хлебникова /кафедра ИиППО Института ИТ РТУ МИРЭА – с. 42, табл. 6, илл.12, ист. 19.

Целью работы является разработка программы для управления деятельностью агентства по подбору.

В рамках работы проведено проектирование программной системы, определены, определены функциональные требования, приведено математическое обеспечение, а также требования к техническому и программному обеспечению.

М. МИРЭА. Ин-т ИТ. Каф. ИиППО. 201\_г. @ Д. Ватулин

Оглавление

[Глава 1. Техническое задание 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682224)

[1.1. Введение 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682225)

[1.1.1. Наименование программы 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682226)

[1.1.2. Назначение и область применения 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682227)

[1.2. Требования к программе или программному изделию 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682228)

[1.2.1. Требования к функциональным характеристикам 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682229)

[1.3. Условия эксплуатации 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682231)

[1.3.1. Требования к составу и параметрам технических средств 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682232)

[1.3.2. Требования к информационной и программной совместимости 5](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682233)

[1.3.2.1. Требования к исходным кодам и языкам программирования 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682234)

[1.3.2.2. Требования к программным средствам, используемым программой 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682235)

[1.4. Требования к программной документации 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682237)

[1.4.1. Предварительный состав программной документации 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682238)

[1.5. Стадии и этапы разработки 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682239)

[1.5.1. Стадии разработки 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682240)

[1.5.2. Этапы разработки 6](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682241)

[1.5.3. Содержание работ по этапам 7](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682242)

[1.6. Порядок контроля и приемки 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682243)

[1.6.1. Виды испытаний 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682244)

[1.6.2. Общие требования к приемке работы 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682245)

[Глава 2. Проектирование и разработка программы 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682246)

[2.1. Проектирование программы 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682247)

[2.1.1. Функциональные требования к программе 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682248)

[2.1.2. Проектирование классов 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682249)

[2.2. Разработка программы 8](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682250)

[2.2.1. Описание проектного решения 9](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682251)

[2.2.2. Тестирование программы 14](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682252)

[Глава 3. Руководство пользователя 15](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682253)

[Заключение 18](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682254)

[Список литературы 19](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682255)

[Приложение №1. Исходный код программы 20](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682256)

[Приложение №2. UML-диаграмма 44](file:///D:\USER\Desktop\Отчёт.docx#_Toc470682257)

**Цель работы:**

Целью работы является разработать программу, для управления деятельностью агентства по подбору персонала. Необходимо разработать программу, для заказа персонала в Агентстве по подбору персонала, с использованием функции поиска нужной услуги по заданным критериям поиска.

# Введение:

В ходе выполнения Курсовой работы должна быть реализована программа для заказа персонала в агентство по подбору персонала, а так же должна быть использована функция поиска нужной услуги по заданным критериям поиска с помощью языка Объективно-Ориентированного Программирования C++. Язык C++ — компилируемый, статически типизированный язык программирования общего назначения. Объектно-ориентированное программирование (ООП) позволяет разложить проблему на составные части, каждая из которых становится самостоятельным объектом. Каждый из объектов содержит свой собственный код и данные, которые относятся к этому объекту.

С объектом связывается набор действий, иначе называемых методами. С точки зрения языка программирования набор действий или методов – это функции, получающие в качестве обязательного параметра указатель на объект и выполняющие определенные действия с данными объекта программирования. Технология ООП запрещает работать с объектом иначе, чем через методы, таким образом, внутренняя структура объекта скрыта от внешнего пользователя.  
  
Описание множества однотипных объектов называется классом. Объект  – это структурированная переменная, содержащая всю информацию о некотором физическом предмете или реализуемом в программе понятии.  
Класс – это описание множества объектов программирования (объектов) и выполняемых над ними действий.

Программное предложение «Программа для заказа персонала в агентство по подбору персонала» представляет собой систему, с помощью которой можно выбрать необходимую услугу и заказать человека, который соответствует вашим критериям поиска.

# Глава 1. Техническое задание

## Введение

### Наименование программы

Наименование программы – «Программа для заказа персонала в агентство по подбору персонала».

### Назначение и область применения

«Программа для управления заказа персонала в агентство по подбору персонала» заказа персонала в агентство по подбору персонала, с использованием функции поиска нужной услуги по заданным критериям поиска.

Требования к программе или программному изделию

### Требования к функциональным характеристикам

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

1. Функцию выбора услуги, которая необходима покупателю.
2. Функции показа всех работников, относящихся к выбранной услуге.
3. Функции выбора подходящего работника.
4. Функции выхода из программы после завершения работы.

## Условия эксплуатации

### Требования к составу и параметрам технических средств

В состав технических средств должен входить IВМ-совместимый персональный компьютер (ПЭВМ), включающий в себя:   
1. Процессор Pentium-2.0Hz, не менее;   
2. Оперативную память объемом, 1 Гигабайт, не менее;   
3. Любую операционную систему;   
4. Стандартный пакет С++;

### Требования к информационной и программной совместимости

#### Требования к исходным кодам и языкам программирования

Исходные коды программы должны быть реализованы на языке C++. В качестве интегрированной среды разработки программы должна быть использована среда Qt.

#### Требования к программным средствам, используемым программой

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией любой операционной системы и ПО C++.

## Требования к программной документации

### Предварительный состав программной документации

Состав программной документации должен включать в себя:

1. Техническое задание;
2. Программу и методики испытаний;
3. Руководство пользователя.

## Стадии и этапы разработки

### Стадии разработки

Разработка должна быть проведена в три стадии:

1. Разработка технического задания;
2. Исследование предметной области;
3. Рабочее проектирование;
4. Внедрение.

### Этапы разработки

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. Разработка программы;
2. Разработка программной документации;
3. Испытания программы.

На стадии внедрения должен быть выполнен этап разработки - подготовка и передача программы.

### Содержание работ по этапам

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. Постановка задачи;
2. Определение и уточнение требований к техническим средствам;
3. Определение требований к программе;
4. Определение стадий, этапов и сроков разработки программы и документации на неё;
5. Согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с требованиями к составу документации.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. Разработка, согласование и утверждение программы и методики испытаний;
2. Проведение приемо-сдаточных испытаний;
3. корректировка программы и программной документации по результатам испытаний.

На этапе подготовки и передачи программы должна быть выполнена работа по подготовке и передаче программы и программной документации в эксплуатацию.

## Порядок контроля и приемки

### Виды испытаний

Демонстрация работы программы и ее функционала.

### Общие требования к приемке работы

Программа должна соответствовать всем вышесказанным требованиям.

# Глава 2. Проектирование и разработка программы

## Проектирование программы

### Функциональные требования к программе

Функциональные требования к программе описаны в пункте № 1.2 Технического задания.

## Проектирование классов

Для реализации проекта будет использован язык программирования C++. Данный язык является одним из самых наиболее распространённых объектно-ориентированных языков программирования (ООП).

Целью данной работы является написание приложения “ Программа для заказа персонала в агентство по подбору персонала ”. Для удобства пользования все действия будут реализованы через кнопки, находящиеся в графическом окне, разработанном при помощи фреймворка Qt.

## Описание проектного решения

**Plumber** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

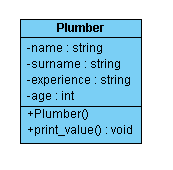


Рис. 1.1 UML диаграмма класса **Plumber**

**Electrician** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

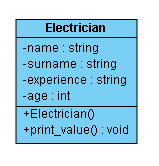
****

Рис. 1.2 UML диаграмма класса **Electrician**

**Driver** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

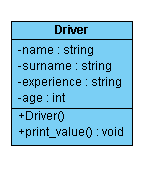
****

Рис. 1.3 UML диаграмма класса **Driver**

**Nanny** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

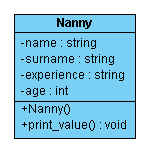
****

Рис. 1.4 UML диаграмма класса **Nanny**

**Tutor** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

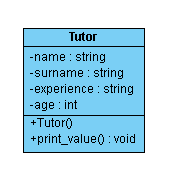
****

Рис. 1.5 UML диаграмма класса **Tutor**

**Waiter** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

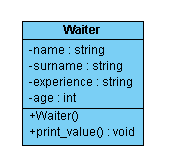
****

Рис. 1.6 UML диаграмма класса **Waiter**

**Designer** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

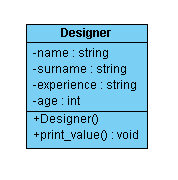
****

Рис. 1.7 UML диаграмма класса **Designer**

**Pharmacist** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

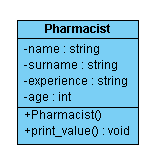
****

Рис. 1.8 UML диаграмма класса **Pharmacist**

**Lawyer** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

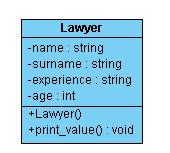
****

Рис. 1.9 UML диаграмма класса **Lawyer**

**Manager** – класс, содержащий имя, фамилию, стаж работы, возраст, поля и методы.

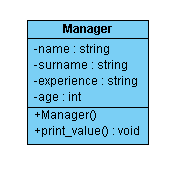
****

Рис. 1.10 UML диаграмма класса **Manager**

# Глава 3. Руководство пользователя

Запуск программы осуществляется через исполняемый файл.exe.

После запуска программы в открытом меню вам потребуется ввести необходимую вам услугу.

Рис.2.1 Скриншот окна “ Меню”.

Если вы хотите посмотреть список всех услуг, необходимо нажать клавишу Tab.

Рис.2.2 Скриншот окна “Список всех услуг ”.

Если вы захотели вернуться в меню нужно снова нажать клавишу Tab.

Рис.2.3 Скриншот окна “Меню”.

После ввода необходимой услуги вы сможете увидеть список работников, относящихся к этой сфере деятельности.

Рис.2.4 Скриншот окна “Список работников”.

Если вы определились с подходящим для вас работником, введите его имя.

Рис.2.5 Скриншот окна “Время прибытия выбранного работника”.

# Заключение

# В результате выполнения курсовой работы была реализована программа «Программа для заказа персонала в агентство по подбору персонала», используя объектно-ориентированное программирование. Написание программы было осуществлено благодаря знаниям, полученным на практических занятиях. Также был усвоен весь теоретический материал.

# 

# Список литературы

1. Зорина Н.В. Методические указания по выполнению курсовой работы для бакалавров обучающихся по направлению 09.03.02 “Информационные системы и технологии”, 09.03.02 “Программная инженерия”
2. Т. А. Павловская. C/C++. Программирование на языке высокого уровня. Из серии "300 лучших учебников".2003 год. 461 стр.
3. Бланшет, Саммерфилд - Qt4 Программирование GUI на С++. 2ed. 2008
4. Шлее Макс - Профессиональное программирование на C++. +CD. Qt 4.8. 2012

# Приложение №1. Исходный код программы

#include <iostream>

#include <string>

#include <stdlib.h>

using namespace std;

class Plumber {

private:

string name;

string surname;

string experience;

int age;

public:Plumber() {}

int a;

Plumber(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Electrician {

string name;

string surname;

string experience;

int age;

public:Electrician() {}

int a;

Electrician(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Driver {

string name;

string surname;

string experience;

int age;

public:Driver() {}

int a;

Driver(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Nanny {

string name;

string surname;

string experience;

int age;

public:Nanny() {}

int a;

Nanny(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Tutor {

string name;

string surname;

string experience;

int age;

public:Tutor() {}

int a;

Tutor(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Waiter {

string name;

string surname;

string experience;

int age;

public:Waiter() {}

int a;

Waiter(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Designer {

string name;

string surname;

string experience;

int age;

public:Designer() {}

int a;

Designer(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Pharmacist {

string name;

string surname;

string experience;

int age;

public:Pharmacist() {}

int a;

Pharmacist(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Lawyer {

string name;

string surname;

string experience;

int age;

public:Lawyer() {}

int a;

Lawyer(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

class Manager {

string name;

string surname;

string experience;

int age;

public:Manager() {}

int a;

Manager(string name, string surname, string experience, int age) {

this->age = age;

this->name = name;

this->surname = surname;

this->experience = experience;

};

void print\_value() {

cout << "Имя: " << name << endl;

cout << "Фамилия: " << surname << endl;

cout << "Стаж работы: " << experience << endl;

cout << "Возраст: " << age << endl;

};

};

int main(int argc, char const\* argv[])

{

string b;

cout << "Здравствуйте!" << endl << "Пожалуйста,введите необходимую вам услугу:" << endl << "Нажмите 1,если вы хотите посмотреть весь список услуг" << endl;

cin >> b;

if (b == 1) {

Plumber ch1((string)"Иван", (string)"Сидоров", (string)"4 года", 32), ch2((string)"Влад", (string)"Кузнецов", (string)"3 года", 25), ch3((string)"Никита", (string)"Воробьев", (string)"0,5 года", 22), ch4((string)"Александр", (string)"Мельников", (string)"10 лет", 45), ch5((string)"Вадим", (string)"Грачев", (string)"6 лет", 32);

ch1.print\_value();

ch2.print\_value();

ch3.print\_value();

ch4.print\_value();

ch5.print\_value();

}

else {

if (b == 2) {

Electrician dh1((string)"Глеб", (string)"Моисеев", (string)"1 год", 24), dh2((string)"Игорь", (string)"Юдин", (string)"4 года", 29), dh3((string)"Евгений", (string)"Краснов", (string)"6 лет", 35), dh4((string)"Дмитрий", (string)"Литвинов", (string)"9 лет", 40), dh5((string)"Алексей", (string)"Бородин", (string)"2 года", 31);

dh1.print\_value();

dh2.print\_value();

dh3.print\_value();

dh4.print\_value();

dh5.print\_value();

}

else {

if (b == 3) {

Driver bh1((string)"Даниил", (string)"Крюков", (string)"5 лет", 30), bh2((string)"Пётр", (string)"Зотов", (string)"2 года", 23), bh3((string)"Владимир", (string)"Шишкин", (string)"10 лет", 38), bh4((string)"Николай", (string)"Самойлов", (string)"1,5 года", 24), bh5((string)"Станислав", (string)"Носов", (string)"15 лет", 43);

bh1.print\_value();

bh2.print\_value();

bh3.print\_value();

bh4.print\_value();

bh5.print\_value();

}

else {

if (b == 4) {

Nanny ah1((string)"Валерия", (string)"Авдеева", (string)"0,5 года", 20), ah2((string)"Евгения", (string)"Головина", (string)"4 года", 26), ah3((string)"Любовь", (string)"Баженова", (string)"18 лет", 56), ah4((string)"Анастасия", (string)"Костина", (string)"4 года", 35), ah5((string)"Александра", (string)"Корнилова", (string)"7 лет", 50);

ah1.print\_value();

ah2.print\_value();

ah3.print\_value();

ah4.print\_value();

ah5.print\_value();

}

else {

if (b == 5) {

Tutor fh1((string)"Галина", (string)"Лукина", (string)"20 лет", 50), fh2((string)"Лилия", (string)"Горелова", (string)"10 лет", 45), fh3((string)"Светлана", (string)"Трифонова", (string)"35 лет", 63), fh4((string)"Андрей", (string)"Зорин", (string)"17 лет", 41), fh5((string)"Михаил", (string)"Антипов", (string)"5 лет", 29);

fh1.print\_value();

fh2.print\_value();

fh3.print\_value();

fh4.print\_value();

fh5.print\_value();

}

else {

if (b == 6) {

Waiter gh1((string)"Ксения", (string)"Кулакова", (string)"1 год", 21), gh2((string)"Кирилл", (string)"Макеев", (string)"0,5 года", 23), gh3((string)"Ольга", (string)"Суханова", (string)"1,5 года", 22), gh4((string)"Дмитрий", (string)"Шубин", (string)"1 год", 20), gh5((string)"Марина", (string)"Русакова", (string)"2 года", 23);

gh1.print\_value();

gh2.print\_value();

gh3.print\_value();

gh4.print\_value();

gh5.print\_value();

}

else {

if (b == 7) {

Designer eh1((string)"Николай", (string)"Успенский", (string)"3 года", 25), eh2((string)"Станислав", (string)"Туманов", (string)"7 лет", 30), eh3((string)"Евгений", (string)"Басов", (string)"0,5 года", 20), eh4((string)"Никита", (string)"Чесноков", (string)"4 года", 26), eh5((string)"Светлана", (string)"Малинина", (string)"1 год", 22);

eh1.print\_value();

eh2.print\_value();

eh3.print\_value();

eh4.print\_value();

eh5.print\_value();

}

else {

if (b == 8) {

Pharmacist mh1((string)"Валентина", (string)"Минаева", (string)"15 лет", 40), mh2((string)"Татьяна", (string)"Денисова", (string)"4 года", 28), mh3((string)"Марьяна", (string)"Измайлова", (string)"8 лет", 32), mh4((string)"Максим", (string)"Пугачев", (string)"1 год", 27), mh5((string)"Валерия", (string)"Проценко", (string)"25 лет", 53);

mh1.print\_value();

mh2.print\_value();

mh3.print\_value();

mh4.print\_value();

mh5.print\_value();

}

else {

if (b == 9) {

Lawyer ph1((string)"Даниил", (string)"Воронков", (string)"6 лет", 33), ph2((string)"Нина", (string)"Сотникова", (string)"2 года", 30), ph3((string)"Дмитрий", (string)"Белкин", (string)"16 лет", 42), ph4((string)"Станислав", (string)"Хромов", (string)"7 лет", 38), ph5((string)"Екатерина", (string)"Анохина", (string)"0,5 года", 25);

ph1.print\_value();

ph2.print\_value();

ph3.print\_value();

ph4.print\_value();

ph5.print\_value();

}

else {

if (b == 10) {

Manager qh1((string)"Кирилл", (string)"Родин", (string)"3 года", 28), qh2((string)"Александра", (string)"Горлова", (string)"6 лет", 35), qh3((string)"Светлана", (string)"Дорохова", (string)"10 лет", 44), qh4((string)"Виктория", (string)"Парамонова", (string)"2,5 года", 27), qh5((string)"Григорий", (string)"Калачев", (string)"12 лет", 38);

qh1.print\_value();

qh2.print\_value();

qh3.print\_value();

qh4.print\_value();

qh5.print\_value();

}

}

}

}

}

}

}

}

}

{

string c;

cout << "Введите имя подходящего для вас работника" << endl;

cin >> c;

if (b == 1) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 2) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 3) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 4) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 5) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 6) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 7) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 8) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 9) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

else {

if (b == 10) {

cout << "Спасибо,что выбрали именно нашу компанию!\n" << c << " сможет выйти на работу в течении следующих 24 часов!" << endl;

}

}

}

}

}

}

}

}

}

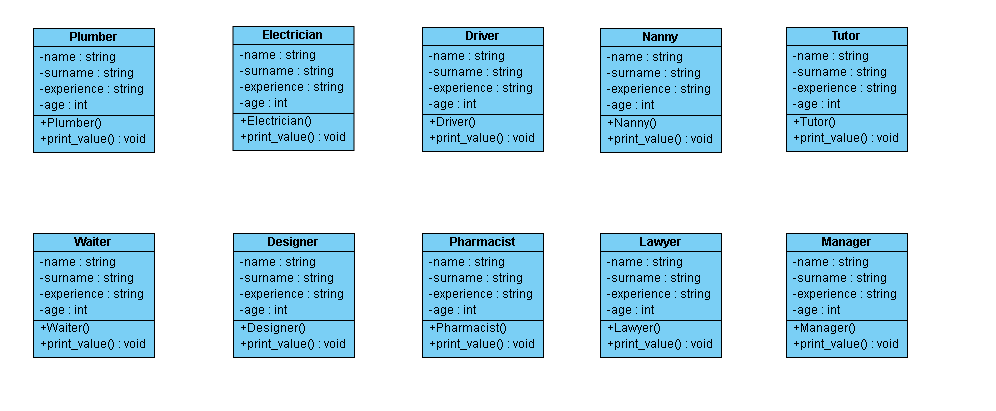
}

}

}

}

# Приложение №2. UML-диаграмма

Рис. 3.1 Общая UML диаграмма