|  |  |
| --- | --- |
| КГЭУ | МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ **Федеральное государственное бюджетное образовательное  учреждение высшего образования** **«КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ»** (ФГБОУ ВО «КГЭУ») |

**Институт цифровых технологий и экономики**

**Кафедра «ИК»**

**Объектно-ориентированное программирование**

**и шаблоны проектирования**

**Отчет о проделанном проекте CASE IN.**

**Итоговая задача**

**Исполнитель**: Мухаметзянов Ильназ

**Группа**: ПМ-1-20

**Вариант №** 7

**Дата выполнения** 15 декабря 2021 г.

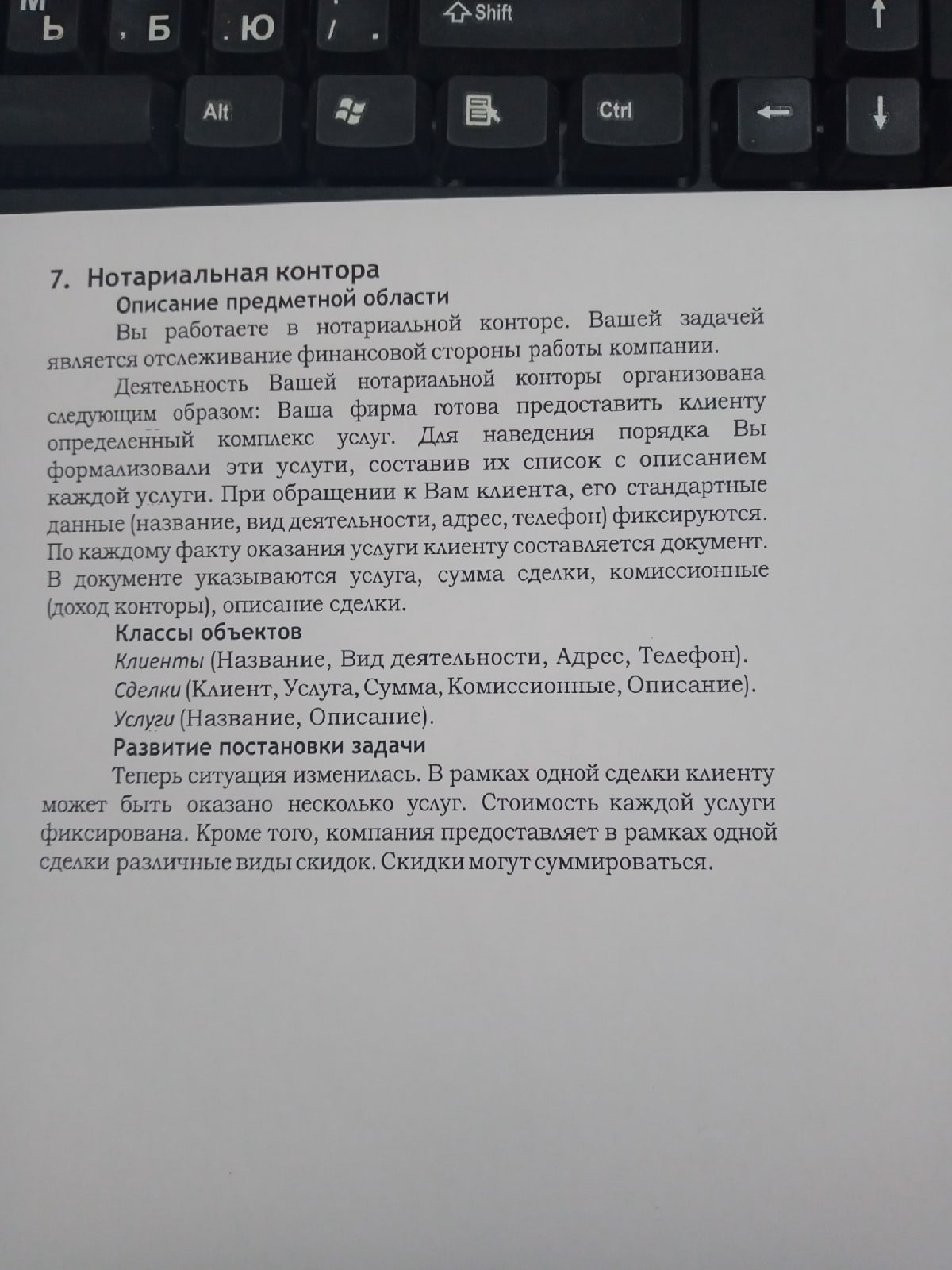
**Дата сдачи** 15.12.2021 **Оценка** \_\_\_\_\_\_\_\_\_\_\_\_

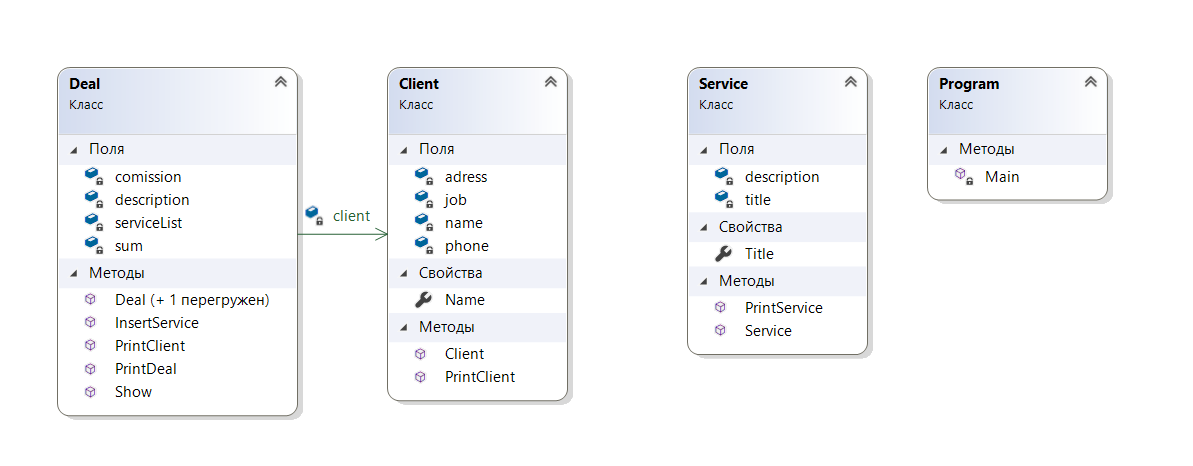
**Подпись преподавателя**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КАЗАНЬ - 2021**

**Задача № 1**

**1. Постановка задачи.**





**2. Листинг программы**

using System;

using System.IO;

using System.Collections.Generic;

namespace ConsoleApp8

{

public class Program

{

static void Main(string[] args)

{

List<Deal> dealList = new List<Deal>();

Deal deal = new Deal(dealList);

bool flag = true;

while (flag)

{

Console.WriteLine("Введите: \n1 - чтобы узнать всех клиентов\n2 - чтобы узнать информацию конкретного клиента\n3 - чтобы добавить услугу клиенту\n4 - выход");

int num = Convert.ToInt32(Console.ReadLine());

switch (num)

{

case (1):

deal.Show(dealList);

break;

case (2):

Console.Write("Введите имя клиента: ");

string name = Console.ReadLine();

deal.PrintClient(dealList, name);

break;

case (3):

Console.WriteLine("Чтобы добавить новую услугу введите имя заказщика");

string name3 = Console.ReadLine();

Console.WriteLine($"Чтобы добавить новую услугу для {name3}a введите название услуги");

Console.WriteLine("Виды услуг: заверение смерти, оплата алиментов, исполнение подписей");

string title = Console.ReadLine();

Console.WriteLine($"Чтобы добавить новую услугу для {name3}a введите описание услуги");

string description = Console.ReadLine();

deal.InsertService(dealList, name3, title, description);

deal.PrintClient(dealList, name3);

break;

case (4):

flag = false;

break;

}

}

}

}

public class Deal

{

decimal sum;

decimal comission;

string description;

Client client;

List<Service> serviceList = new List<Service>();

public Deal(string clientName, string clientJob, string clientAdress, string clientPhone, string serviceTitle, string serviceDescription, decimal sum, decimal comission, string description)

{

client = new Client(clientName, clientJob, clientAdress, clientPhone);

serviceList.Add(new Service(serviceTitle, serviceDescription));

this.sum = sum;

this.comission = comission;

this.description = description;

}

public Deal(List<Deal> deals)

{

string job = string.Empty;

string adress = string.Empty;

string phone = string.Empty;

string title = string.Empty;

string description = string.Empty;

StreamReader sr = new StreamReader("a.txt");

while (true)

{

try

{

string name = sr.ReadLine();

if (name == null)

break;

else

{

job = sr.ReadLine();

adress = sr.ReadLine();

phone = sr.ReadLine();

title = sr.ReadLine();

description = sr.ReadLine();

sum = Convert.ToDecimal(sr.ReadLine());

comission = Convert.ToDecimal(sr.ReadLine());

this.description = sr.ReadLine();

deals.Add(new Deal(name, job, adress, phone, title, description, sum, comission, this.description));

}

}

catch

{

Console.WriteLine("Нет файла");

}

}

}

public void InsertService(List<Deal> deals, string name, string serviceTitle, string serviceDescription)

{

for (int i = 0; i < deals.Count; i++)

{

if (deals[i].client.Name == name)

deals[i].serviceList.Add(new Service(serviceTitle, serviceDescription));

}

for (int i = 0; i < deals.Count; i++)

{

for (int j = 0; j < deals[i].serviceList.Count; j++)

{

if (deals[i].serviceList[j].Title == serviceTitle && j != 0)

{

switch (deals[i].serviceList[j].Title)

{

case ("заверение смерти"):

deals[i].sum += 1000;

break;

case ("исполнение подписей"):

deals[i].sum += 500;

break;

case ("оплата алиментов"):

deals[i].sum += 1500;

break;

}

}

}

}

}

public void PrintDeal(List<Deal> deals, string name)

{

for (int i = 0; i < deals.Count; i++)

{

if (deals[i].client.Name == name)

{

if (deals[i].serviceList.Count > 2)

{

deals[i].sum = deals[i].sum \* (decimal)0.5;

Console.WriteLine($"\tСумма сделки - {sum}, комисионные - {comission}, описание - {description}, ваша скидка составила 50% от общей суммы");

}

if (deals[i].serviceList.Count > 1)

{

deals[i].sum = deals[i].sum \* (decimal)0.9;

Console.WriteLine($"\tСумма сделки - {sum}, комисионные - {comission}, описание - {description}, ваша скидка составила 10% от общей суммы");

}

else

Console.WriteLine($"\tСумма сделки - {sum}, комисионные - {comission}, описание - {description}");

break;

}

}

}

public void PrintClient(List<Deal> deals, string name)

{

for (int i = 0; i < deals.Count; i++)

{

if (name == deals[i].client.Name)

{

deals[i].client.PrintClient();

for (int j = 0; j < deals[i].serviceList.Count || j == 0; j++)

{

deals[i].serviceList[j].PrintService();

}

deals[i].PrintDeal(deals, name);

break;

}

}

}

public void Show(List<Deal> deals)

{

for (int i = 0; i < deals.Count; i++)

{

deals[i].client.PrintClient();

for (int j = 0; j < deals[i].serviceList.Count || j == 0; j++)

{

deals[i].serviceList[j].PrintService();

}

deals[i].PrintDeal(deals, deals[i].client.Name);

}

}

}

public class Client

{

string name;

string job;

string adress;

string phone;

public Client(string name, string job, string adress, string phone)

{

this.name = name;

this.job = job;

this.adress = adress;

this.phone = phone;

}

public string Name { get { return name; } }

public void PrintClient()

{

Console.WriteLine($"{name,2} : Вид деятельности - {job}, Адресс - {adress}, Телефон - {phone}");

}

}

public class Service

{

string title;

string description;

public string Title { get { return title; } }

public Service(string title, string description)

{

this.description = description;

this.title = title;

}

public void PrintService()

{

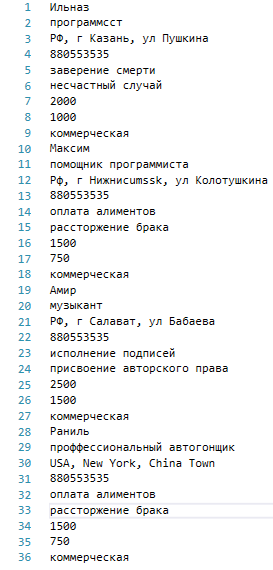
Console.WriteLine($"\tНазвание услуги - {title}, описание - {description}");

}

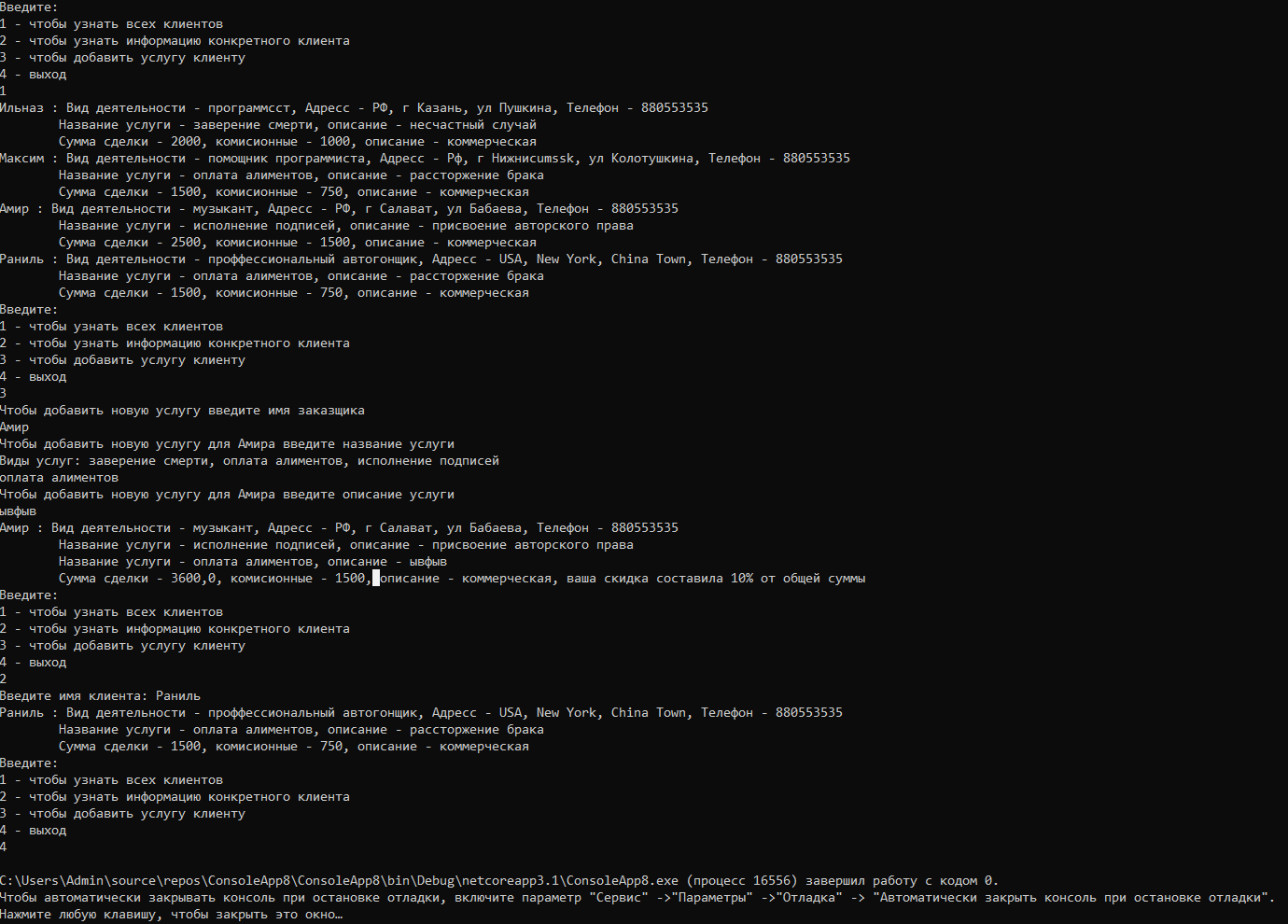
}

}

**3. Исходные данные**



**4. Результаты работы программы.**



**Подпись исполнителя**

