МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧЕРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

КАФЕДРА «ИИТ»

ОТЧЕТ

по лабораторной работе №4

по дисциплине «СПП»

Тема: «ООП в JAVA»

**Выполнила:**Андросюк Мария

**Группа:** ПО-5

**Преподаватель:**

Крощенко А.А.

Брест 2021

Вариант 1

**Цель работы**: приобрести практические навыки в области объектно-ориентированного проектирования.

**Ход работы:**1) Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы.

Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

Создать класс Account (счет) с внутренним классом, с помощью объектов которого можно хранить информацию обо всех операциях со счетом (снятие, платежи, поступления).

**Код программы:**

public class Account

{

private class owner

{

int removals;

int payments;

int entrances;

@Override

public String toString()

{

return "removals "+removals+"$\n"+"payments "+payments+"$\n"+"entrances "+entrances+"$\n";

}

}

owner own = new owner();

public void Addowner (int removals, int payments, int entrances)

{

own.removals = removals;

own.payments = payments;

own.entrances = entrances;

}

public void show()

{

System.out.println(own.toString());

}

public void removeMoney(int rem)

{

own.removals += rem;

}

public void pay(int pay)

{

own.payments += pay;

}

public void entranc(int ent)

{

own.entrances += ent;

}

}

public class Lab4

{

public static void main(String[] args)

{

Account ac = new Account();

ac.Addowner(200, 371, 3423);

ac.show();

ac.removeMoney(100);

ac.pay(800);

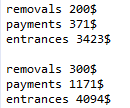
ac.entranc(671);

ac.show();

}

}

**Результат работы программы:**



**2)** Реализовать агрегирование. При создании класса агрегируемый класс объявляется как атрибут (локальная переменная, параметр метода). Включить в каждый класс 2-3 метода на выбор. Продемонстрировать использование разработанных классов.

Создать класс Страница, используя класс Абзац.

**Код программы:**

import java.util.\*;

public class Page

{

private class paragraph

{

String par;

@Override

public String toString()

{

return par;

}

}

Vector<paragraph> pars = new Vector<>();

public void addPar(String str)

{

paragraph parag = new paragraph();

parag.par = str;

pars.add(parag);

}

public void showPage()

{

for(paragraph x:pars)

{

System.out.println("\t"+x);

}

}

}

public class Lab4\_2

{

public static void main(String[] args)

{

Page pg = new Page();

pg.addPar("Gadhasdkfdlfkhjklgfjklhgfkldkljkljghdss\ndfjfkjghsdfkjlhgdffjklfdkjldfjklkghdfkjlghsdfkl\ndfldfdffdkldf");

pg.addPar("Lasfhdgfdfhgfhfghsdfkljklsdjldfksjlsdfj\nfhdfghdfjjkhjklhklhkljhjklkljhklhljkhkljhjklhhg\nfhdhkdfjfkdj");

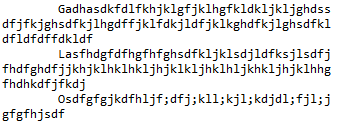
pg.addPar("Osdfgfgjkdfhljf;dfj;kll;kjl;kdjdl;fjl;jjgfgfhjsdf");

pg.showPage();

}

}

**Результат работы программы:**



**3)** Система Больница. Пациенту назначается лечащий Врач. Врач может сделать назначение Пациенту (процедуры, лекарства, операции). Медсестра или другой Врач выполняют назначение. Пациент может быть выписан из Больницы по окончании лечения, при нарушении режима или иных обстоятельствах.

**Код программы:**

import java.util.\*;

public class hospital

{

private class doctor

{

String docName;

String patName;

@Override

public String toString()

{

return "Patient name: "+patName+"\nProcedure: "+"\nDoctor: "+docName+"\n";

}

}

private class patient

{

String patName;

String docName;

String proc;

String condition;

int age;

@Override

public String toString()

{

return "Patient name: "+patName+"\nProcedure: "+proc+"\nAge: "+age+"\nDoctor: "+docName+"\n"+"Condition: "+condition+"\n";

}

public boolean equals(patient pat, String patName)

{

if (pat.patName == patName) return true;

else return false;

}

}

public void addPatient(String patName, int age)

{

patient pat = new patient();

pat.patName = patName;

pat.age = age;

pat.condition = "In hospital";

pats.add(pat);

}

public void addDoc(String docName, String patName)

{

doctor doc = new doctor();

doc.docName = docName;

doc.patName = patName;

docs.add(doc);

for(patient x: pats)

{

if (x.equals(x, patName)) x.docName = docName; else System.out.println("Have no patient with this name in our hospital");

}

}

public void addProc(String proc, String patName, String docName)

{

boolean has = false;

for(doctor d: docs)

if (d.docName == docName)

{

has = true;

if(d.patName == patName)

{

for(patient p: pats)

{

if (p.equals(p, patName))

{

p.proc = proc;

}

}

}

break;

}

if (has == false)

{

System.out.println("We haven't this doctor!");

return;

}

}

public void show()

{

for(patient x: pats)

{

System.out.println(x.toString());

}

}

public void leavePat(String patName)

{

for(patient p: pats)

{

if(p.equals(p, patName))

{

p.condition = "Discharged";

}

}

}

Vector<patient> pats = new Vector<>();

Vector<doctor> docs = new Vector<>();

}

public class Lab4\_3

{

public static void main(String[] args)

{

hospital hosp = new hospital();

hosp.addPatient("Vasya", 15);

hosp.show();

hosp.addDoc("Grysha", "Vasya");

hosp.show();

hosp.addProc("Operation", "Vasya", "Grysha");

hosp.show();

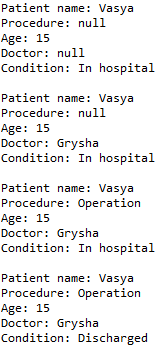
hosp.leavePat("Vasya");

hosp.show();

}

}

**Результат работы программы:**



**Вывод:** приобрела практические навыки в области объектно-ориентированного проектирования.