# Лабораторная работа № 6

**Тема работы: «Разработка программ, реализующих механизм**

**наследования»**

**Цель работы:** закрепить навык наследования

**Задание:** реализовать классы по заданию (в задачах второго уровня должно присутствовать не менее 4 классов). Наследование применяется в логически нужных местах.

В классе должны быть реализованы:

* поля (закрытые или защищённые), нужное количество;
* свойства (для каждого поля);
* конструкторы (не менее двух);
* наследование конструкторов;
* методы (не менее трех);
* переопределенные родительские методы (нужные по логике);
* метод toString().

Номер варианта соответствует вашему номеру по списку.

**Задание 1**

1. Создать объект класса Текст, используя классы Предложение, Слово. Методы: дополнить текст, вывести на консоль текст, заголовок текста.
2. Создать объект класса Автомобиль, используя классы Колесо, Двигатель. Методы: ехать, заправляться, менять колесо, вывести на консоль марку автомобиля.
3. Создать объект класса Самолет, используя классы Крыло, Шасси, Двигатель. Методы: летать, задавать маршрут, вывести на консоль маршрут.
4. Создать объект класса Государство, используя классы Область, Район, Город. Методы: вывести на консоль столицу, количество областей, площадь, областные центры.
5. Создать объект класса Планета, используя классы Материк, Океан, Остров. Методы: вы вести на консоль название материка, планеты, количество материков.
6. Создать объект класса Звездная система, используя классы Планета, Звезда, Луна. Методы: вывести на консоль количество планет в звездной системе, название звезды, добавление планеты в систему.
7. Создать объект класса Компьютер, используя классы Винчестер, Дисковод, Оперативная память, Процессор. Методы: включить, выключить, проверить на вирусы, вывести на консоль размер винчестера.
8. Создать объект класса Квадрат, используя классы Точка, Отрезок. Методы: задание размеров, растяжение, сжатие, поворот, изменение цвета.
9. Создать объект класса Круг, используя классы Точка, Окружность. Методы: задание размеров, изменение радиуса, определение принадлежности точки данному кругу.
10. Создать объект класса Щенок, используя классы Животное, Собака. Методы: вывести на консоль имя, подать голос, прыгать, бегать, кусать.
11. Создать объект класса Наседка, используя классы Птица, Кукушка. Методы: летать, петь, нести яйца, высиживать птенцов.
12. Создать объект класса Текстовый файл, используя классы Файл, Директория. Методы: создать, переименовать, вывести на консоль содержимое, дополнить, удалить.
13. Создать объект класса Одномерный массив, используя классы Массив, Элемент. Методы: создать, вывести на консоль, выполнить операции (сложить, вычесть, перемножить).
14. Создать объект класса Простая дробь, используя класс Число. Методы: вывод на экран, сложение, вычитание, умножение, деление.
15. Создать объект класса Дом, используя классы Окно, Дверь. Методы: закрыть на ключ, вывести на консоль количество окон, дверей.
16. Создать объект класса Цветок, используя классы Лепесток, Бутон.

Методы: расцвести, завять, вывести на консоль цвет бутона.

1. Создать объект класса Дерево, используя классы Лист, Ветка. Методы: зацвести, опасть листьям, покрыться инеем, пожелтеть листьям.
2. Создать объект класса Пианино, используя классы Клавиша, Педаль.

Методы: настроить, играть на пианино, нажимать клавиши.

**Задание 2**

1. Цветочница. Определить иерархию цветов. Создать несколько объектов- цветов. Собрать букет (используя аксессуары) с определением его стоимости. Провести сортировку цветов в букете на основе уровня свежести. Найти цветок в букете, соответствующий заданному диапазону длин стеблей.
2. Новогодний подарок. Определить иерархию конфет и прочих сладостей. Создать несколько объектов-конфет. Собрать детский подарок с определением его веса. Провести сортировку конфет в подарке на основе одного из параметров. Найти конфету в подарке, соответствующую заданному диапазону содержания сахара.
3. Домашние электроприборы. Определить иерархию электроприборов. Включить некоторые в розетку. Подсчитать потребляемую мощность. Провести сортировку приборов в квартире на основе мощности. Найти прибор в квартире, соответствующий заданному диапазону параметров.
4. Шеф-повар. Определить иерархию овощей. Сделать салат. Подсчитать калорийность. Провести сортировку овощей для салата на основе одного из параметров. Найти овощи в салате, соответствующие заданному диапазону калорийности.
5. Звукозапись. Определить иерархию музыкальных композиций. Записать на диск сборку. Подсчитать продолжительность. Провести перестановку композиций диска на основе принадлежности к стилю. Найти композицию, соответствующую заданному диапазону длины треков.
6. Камни. Определить иерархию драгоценных и полудрагоценных камней. Отобрать камни для ожерелья. Подсчитать общий вес (в каратах) и стоимость. Провести сортировку камней ожерелья на основе ценности. Найти камни в ожерелье, соответствующие заданному диапазону параметров прозрачности.
7. Мотоциклист. Определить иерархию амуниции. Экипировать мотоциклиста. Подсчитать стоимость. Провести сортировку амуниции на основе веса. Найти элементы амуниции, соответствующие заданному диапазону параметров цены.
8. Транспорт. Определить иерархию подвижного состава железнодорожного транспорта. Создать пассажирский поезд. Подсчитать общую численность пассажиров и багажа. Провести сортировку вагонов поезда на основе уровня комфортности. Найти в поезде вагоны, соответствующие заданному диапазону параметров числа пассажиров.
9. Авиакомпания. Определить иерархию самолетов. Создать авиакомпанию. Посчитать общую вместимость и грузоподъемность. Провести сортировку самолетов компании по дальности полета. Найти самолет в компании, соответствующий заданному диапазону параметров потребления горючего.
10. Таксопарк. Определить иерархию легковых автомобилей. Создать таксопарк. Подсчитать стоимость автопарка. Провести сортировку автомобилей парка по расходу топлива. Найти автомобиль в компании, соответствующий заданному диапазону параметров скорости.
11. Страхование. Определить иерархию страховых обязательств. Собрать из обязательств дериватив. Подсчитать стоимость. Провести сортировку обязательств в деривативе на основе уменьшения степени риска. Найти обязательство в деривативе, соответствующее заданному диапазону параметров.
12. Мобильная связь. Определить иерархию тарифов мобильной компании. Создать список тарифов компании. Подсчитать общую численность клиентов. Провести сортировку тарифов на основе размера абонентской платы. Найти тариф в компании, соответствующий заданному диапазону параметров. Фургон кофе. Загрузить фургон определенного объема грузом на определенную сумму из различных сортов кофе, находящихся, к тому же, в разных физических состояниях (зерно, молотый, растворимый в банках и пакетиках). Учитывать объем кофе вместе с упаковкой. Провести сортировку товаров на основе соотношения цены и веса. Найти в фургоне товар, соответствующий заданному диапазону параметров качества.
13. Игровая комната. Подготовить игровую комнату для детей разных возрастных групп. Игрушек должно быть фиксированное количество в пределах выделенной суммы денег. Должны встречаться игрушки родственных групп: маленькие, средние и большие машины, куклы, мячи, кубики. Провести сортировку игрушек в комнате по одному из параметров. Найти игрушки в комнате, соответствующие заданному диапазону параметров.
14. Налоги. Определить множество и сумму налоговых выплат физического лица за год с учетом доходов с основного и дополнительного мест работы, авторских вознаграждений, продажи имущества, получения в подарок денежных сумм и имущества, переводов из-за границы, льгот на детей и материальной помощи. Провести сортировку налогов по сумме.
15. Счета. Клиент может иметь несколько счетов в банке. Учитывать возможность блокировки/разблокировки счета. Реализовать поиск и сортировку счетов. Вычисление общей суммы по счетам. Вычисление суммы по всем счетам, имеющим положительный и отрицательный балансы отдельно.
16. Туристические путевки. Сформировать набор предложений клиенту по выбору туристической путевки различного типа (отдых, экскурсии, лечение, шопинг, круиз и т. д.) для оптимального выбора. Учитывать возможность выбора транспорта, питания и числа дней. Реализовать выбор и сортировку путевок.
17. Кредиты. Сформировать набор предложений клиенту по целевым кредитам различных банков для оптимального выбора. Учитывать возможность досрочного погашения кредита и/или увеличения кредитной линии. Реализовать выбор и поиск кредита.

**Оснащение работы**

Задание по варианту, ЭВМ, среда разработки **IntelliJ IDEA**

**Основные теоретические сведения**

**Наследование**

Наследование является неотъемлемой частью Java. При использовании

наследования вы говорите: Этот новый класс похож на тот старый класс. В коде это пишется как **extends**, после которого указываете имя базового класса. Тем самым вы получаете доступ ко всем полям и методам базового класса. Используя наследование, можно создать общий класс, которые определяет характеристики, общие для набора связанных элементов. Затем вы можете наследоваться от него и создать новый класс, который будет иметь свои уникальные характеристики. Главный наследуемый класс в Java называют суперклассом. Наследующий класс называют подклассом. Получается, что подкласс - это специализированная версия суперкласса, которая наследует все члены суперкласса и добавляет свои собственные уникальные элементы. К примеру, в Android есть класс **View** и подкласс **TextView**.

Чтобы наследовать класс, достаточно вставить имя наследуемого класса с использованием ключевого слова **extends**:

public class MainActivity extends Activity {

}

В этом коде мы наследуемся от класса **Activity** и добавляем свой код, который будет отвечать за наше приложение.

Подкласс в свою очередь может быть суперклассом другого подкласса. Так например, упоминавший ранее класс**TextView** является суперклассом для **EditText**.

В производный класс можно добавлять новые методы.

Для каждого создаваемого подкласса можно указывать только один суперкласс. При этом никакой класс не может быть собственным суперклассом. Хотя подкласс включает в себя все члены своего суперкласса, он не может получить доступ к тем членам суперкласса, которые объявлены

как **private**.

Помните, мы создавали класс **Box** для коробки кота. Давайте наследуемся от этого класса и создадим новый класс, который будет иметь не только размеры коробки, но и вес.

В том же файле **Box.java** после последней закрывающей скобки добавьте новый код:

class HeavyBox extends Box { int weight; // вес коробки

// конструктор

HeavyBox(int w, int h, int d, int m) {

width = w; height = h; depth = d;

weight = m; // масса

}

}

Возвращаемся в главную активность и пишем код: HeavyBox box = new HeavyBox(15, 10, 20, 5); int vol = box.getVolume();

mInfoTextView.setText("Объѐм коробки: " + vol + " Вес коробки: " + box.weight);

Обратите внимание, что мы вызываем метод **getVolume()**, который не

прописывали в классе **HeavyBox**. Однако мы можем его использовать, так как мы наследовались от класса **Box** и нам доступны все открытые поля и методы. Заодно мы вычисляем вес коробки с помощью новой переменной, которую добавили в подкласс.

Теперь у нас появилась возможность складывать в коробку различные

вещи.

При желании вы можете создать множество разных классов на основе

одного суперкласса. Например, мы можем создать цветную коробку.

class ColorBox extends Box { int color; // цвет коробки

// конструктор

ColorBox(int w, int h, int d, int c) { width = w;

height = h; depth = d;

color = c; // цвет

}

}

**Ключевое слово super**

В Java существует ключевое слово *super*, которое обозначает **суперкласс**, т.е. класс, производным от которого является текущий класс. В данном случае, супер не означает превосходство, скорее даже наоборот, дочерний класс имеет больше методов, чем родительский. Само слово пошло из теории множеств, где используется термин супермножество. Посмотрим, зачем это нужно.

В конструкторе **HeavyBox** мы дублировали поля **width**,**height** и **depth**, которые уже есть в классе **Box**. Это не слишком эффективно. Кроме того, возможны ситуации, когда суперкласс имеет закрытые члены данных, но мы

хотим иметь к ним доступ. Через наследование это не получится, так как закрытые члены класса доступны только родному классу. В таких случаях вы можете сослаться на суперкласс.

Ключевое слово **super** можно использовать для вызова конструктора суперкласса и для обращения к члену суперкласса, скрытому членом подкласса.