СИСТЕМЫ И СРЕДЫ ПРОГРАММИРОВАНИЯ

Требования по выполнению лабораторных работ

Каждая лабораторная работа должна быть выполнена самостоятельно.

При защите лабораторной работы необходимо предоставить:

- Отчет (титульный лист, цель работы, вариант задания на работу, исходный код программы);
- Исходные тексты программы (в электронном виде, на электронную почту maks.gek@gmail.com);

Тексты программ должны быть оформлены в соответствии с принятыми стандартами написания кода на Java.

Тексты должны содержать комментарии разработчика для всех методов и классов.

Если перечисленные выше требования не выполнены – **работа не принимается.**

Лабораторная работа №1

Объектная модель, пакеты

Цель: изучение синтаксиса языка Java, объектной модели, общей структуры программ.

ЗАДАНИЕ

Классы должны быть объявлены в пакете **ru.bstu.iitus.kb31.xxx**, где **xxx** – Ваши инициалы.

Вариант 1

Создать абстрактный класс Construction (сооружение)

public abstract void init(Scanner scanner) // считывание параметров с консоли

public int getExploitationPeriod(); // возвращает срок эксплуатации сооружения

public String toString() // возвращается состояние объекта в виде строки

// (определяется только в наследниках, т.к. определен в

// Object)

Построить иерархию классов:

Сооружение → Здание → Супермаркет

→ Частный дом

→ Многоквартирный дом

→ Путепроводное сооружение → Мост

→ Туннель

Написать программу, которая:

1) Считывает с консоли количество сооружений.

2) В цикле считывает параметры. Сначала спрашивается тип сооружения и создается объект нужного класса.
Затем у объекта вызывается метод init() и вводятся характеристики объекта (основной материал, из которого сделано сооружение, срок экплуатации и т.д.).
Метод init() разный у разных классов.

3) Считанные объекты помещаются в массив.

4) Ищется сооружение с минимальным сроком эксплуатации и выводится на экран (вывод через toString())

Вариант 2

Создать абстрактный класс SportsEquipment (спортивный инвентарь)

public abstract void init(Scanner scanner) // считывание параметров с консоли

public int getSportType(); // возвращает вид спорта, к которому относится

public String toString() // возвращается состояние объекта в виде строки

// (определяется только в наследниках, т.к. определен в

// Object)

Построить иерархию классов:

Спортивный инвентарь → Мяч → Волейбольный мяч

→ Теннисный мяч

→ Ракетка

→ Метательное копье

→ Тренажерный → Штанга

→ Гиря

Написать программу, которая:

1) Считывает с консоли количество предметов инвентаря.

2) В цикле считывает параметры. Сначала спрашивается тип инвентаря и создается объект нужного класса.
Затем у объекта вызывается метод init() и вводятся характеристики объекта (вид спорта, к которому относится, характеристики: для мяча — радиус, для гири и штанги — вес и т.д.).
Метод init() разный у разных классов.

3) Считанные объекты кладутся в массив.

4) Ищется весь инвентарь, относящийся к теннису, и выводится на экран (вывод через toString())

Вариант 3

Создать абстрактный класс Person (человек) с методами:

public abstract void init(Scanner scanner) // считывание параметров с консоли

public int getAge(); // возвращается возраст человека на текущий момент

// (полное количество лет)

public String toString() // возвращается состояние объекта в виде строки
// (определяется только в наследниках, т.к.
определен в
// Object)

Построить иерархию классов:

Человек → Студент
→ Школьник
→ Сотрудник → Преподаватель
→ Директор

Написать программу, которая:

1) Считывает с консоли количество человек.

2) В цикле считывает параметры. Сначала спрашивается тип персоны и создается объект нужного класса. Затем у объекта вызывается метод init() и вводятся характеристики объекта (номер зачетной книжки для студента, номер сертификата для преподавателя и т.д.).

Метод init() разный у разных классов.

3) Считанные объекты кладутся в массив.

4) Ищется самый младший человек и выводится на экран (вывод через toString())

Вариант 4

Создать абстрактный класс Product с методами:

public abstract void init(Scanner scanner) // считывание параметров с консоли

public abstract int getCost() // возвращает стоимость товара

public abstract boolean canBuy(int cost); // определяет, можно ли купить товар за имеющуюся // сумму

public String toString() // возвращается состояние объекта в виде строки
// (определяется только в наследниках, т.к.
определен в // Object)

Построить иерархию классов:

Товар → Игрушка → Кубик-рубик
→ Молочный → Сыр
→ Техника → Камера
→ Телевизор

Написать программу, которая:

1) Считывает с консоли количество товаров.

2) В цикле считывает параметры. Сначала спрашивается вид товара и создается объект нужного класса. Затем у объекта вызывается метод init() и вводятся характеристики

объекта (стоимость, наименование игрушки, разрешение камеры, размер диагонали телевизора и т.д.).
Метод init() разный у разных классов.

3) Считанные объекты кладутся в массив

4) Ищется самый дешевый товар и выводится на экран (вывод через toString())

Вариант 5

Создать абстрактный класс Construction (сооружение)

public abstract void init(Scanner scanner) // считывание параметров с консоли

public int getConstructionCost(); // возвращает стоимость возведения сооружения

public String toString() // возвращается состояние объекта в виде строки
// (определяется только в наследниках, т.к.
// определен в классе Object)

Построить иерархию классов:

Сооружение → Здание → Отель
→ Ресторан
→ Дом → Многоквартирный дом
→ Частный дом

Написать программу, которая:

1) Считывает с консоли количество сооружений.

2) В цикле считывает параметры. Сначала спрашивается тип сооружения и создается объект нужного класса.
Затем у объекта вызывается метод init() и вводятся характеристики объекта (основной материал, из которого сделано сооружение, стоимость возведения сооружения и т.д.).
Метод init() разный у разных классов.

3) Считанные объекты помещаются в массив.

4) Массив сооружений сортируется по стоимости возведения и в отсортированном виде выводится на экран (вывод через toString())

Вариант 6

Создать абстрактный класс SportsEquipment (спортивный инвентарь)

public abstract void init(Scanner scanner) // считывание параметров с консоли

public int getSportType(); // возвращает вид спорта, к которому относится

public String toString() // возвращается состояние объекта в виде строки
// (определяется только в наследниках, т.к.
определен в
// Object)

Построить иерархию классов:

Спортивный инвентарь → Мяч → Баскетбольный мяч
→ Теннисный мяч
→ Ракетка

→ Ядро

→ Тренажерный → Штанга

→ Гиря

Написать программу, которая:

1) Считывает с консоли количество предметов инвентаря.

2) В цикле считывает параметры. Сначала спрашивается тип инвентаря и создается объект нужного класса.
Затем у объекта вызывается метод init() и вводятся характеристики объекта (вид спорта, к которому относится, характеристики: для мяча — радиус, для гири, ядра и штанги — вес и т.д.). Метод init() разный у разных классов.

3) Считанные объекты кладутся в массив.

4) Ищется весь инвентарь, имеющий вес как основную характеристику, и выводится на экран (вывод через toString())

Вариант 7

Создать абстрактный класс Person (человек) с методами:

public abstract void init(Scanner scanner) // считывание параметров с консоли

public int getWeight(); // возвращается вес человека на текущий момент

// (полное количество лет)

public String toString() // возвращается состояние объекта в виде строки

// (определяется только в наследниках, т.к. определен в

// Object)

Построить иерархию классов:

Человек → Студент

→ Пенсионер

→ Сотрудник → Преподаватель

→ Секретарь

Написать программу, которая:

1) Считывает с консоли количество человек.

2) В цикле считывает параметры. Сначала спрашивается тип персоны и создается объект нужного класса. Затем у объекта вызывается метод init() и вводятся характеристики объекта (номер зачетной книжки для студента, номер сертификата для преподавателя и т.д.).

Метод init() разный у разных классов.

3) Считанные объекты кладутся в массив.

4) Ищется самый легкий человек и выводится на экран (вывод через toString())

5)

Вариант 8

Создать абстрактный класс Product с методами:

```
public abstract void init(Scanner scanner)    // считывание параметров с консоли
public abstract int getCost()                 // возвращает стоимость товара
public abstract boolean canBuy(int cost);     // определяет, можно ли купить товар за имеющуюся
                                                  // сумму
public String toString()                      // возвращается состояние объекта в виде строки
                                                  // (определяется только в наследниках, т.к. определен в
                                                  // Object)
```

Построить иерархию классов:

Товар → Игрушка → Лего

→ Молочный → Молоко

→ Кефир

→ Техника → Камера

→ Ноутбук

Написать программу, которая:

1) Считывает с консоли количество товаров.

2) В цикле считывает параметры. Сначала спрашивается вид товара и создается объект нужного класса. Затем у объекта вызывается метод init() и вводятся характеристики объекта (стоимость, наименование игрушки, разрешение камеры, размер экрана ноутбука и т.д.).
Метод init() разный у разных классов.

3) Считанные объекты кладутся в массив

4) Ищется самый дорогой товар и выводится на экран (вывод через toString())