КИРИЛЛОВ Д.В.

ПРАКТИКУМ ПО КУРСУ “ОСНОВЫ JAVA-ТЕХНОЛОГИЙ”

Задачи и упражнения

Самара, 2007

В пособии приведены сгруппированные по занятиям упражнения по программированию, задачи и контрольные вопросы, предназначенные для решения студентами, изучающими курс “Основы Java-технологий”. Для каждого практического занятия определена область изучаемых Java-технологий и цели практического занятия. Пособие рекомендуется как для студентов изучающих курс языка Java, так и преподавателей, проводящих по этому курсу практические занятия.

Практическое занятие №1

Основы Языка Java, массивы, примитивные типы, объявление классов

1. Цели работы

Целями практического занятия являются**:**

- **з**накомство с основными синтаксическими конструкциями языка Java;
- изучение структуры консольного приложения на языке Java;
- работа со стандартными потоками ввода/вывода;
- получение навыков работы с командной строкой;

Предметом изучения являются методы и свойства следующих классов платформы J2SE**:**

- классы-оболочки над примитивными типами java.lang.Integer,java.lang.Double, java.lang.Float;
- класс объяляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;
- классы, предназначенные для работы с потоками ввода-вывода - java.io.InputStreamReader, java.io.InputStream, java.io.PrintStream

2. Упражнения на программирование

1. Реализовать приложение, для введенной с консоли последовательности целых чисел вычисляющее следующие функции- нахождение среднего значения, максимального значения, минимального значения, произведения, суммы, разности значений. В качестве объекта, применяемого для хранения последовательности целых чисел, использовать массив. Методы, реализующие данные функции должны иметь прототипы следующего вида:

```
public static int getSumNumbers(int[] numbers);
```

2. Дополнить приложение из упражнения 1, методами, реализующими аналогичные операции над числами типа Double, перегружающие одноименные методы реализованные в упражнении 1, с функ-

цией проверки вводимых значений на соответствие тому или иному типу. Проверку выполнять в методе main приложения. В случае если хотя бы одно из значений не соответствует одному из заданных типов, в консоль должно быть выведено сообщение об ошибке и программа должна завершить свою работу.

3. Изменить приложение реализованное в упражнение 2 таким образом, чтобы последовательность чисел передавалась как параметр командной строки.

4. Спроектировать и реализовать класс для обработки исключительной ситуации, возникающей в том случае, если элемент последовательности не является числом, с возможностью представления информации об элементе, обработка которого вызвала исключительную ситуацию. Изменить приложение, таким образом, чтобы метод разбора последовательности обладал возможностью генерировать исключительную ситуацию являющуюся объектом заданного класса.

5. Дополнить приложение таким образом, чтобы имелась возможность формирования исходной числовой последовательности состоящих из чисел сформированных генератором случайных чисел.

6. Реализовать класс, в который выделены методы, реализованные в пунктах 1-5.

3. Задачи

1. Проанализируйте следующий код метода. Что произойдет в результате его вызова?

```
public void doSomething(){
      int x;
      double y;
      x=(int)Math.random();
            y=Math.ceil((double)
            (int)Math.random());
      do{
         System.out.println(x/y);
      } while (x<y);
                              }
```

2. Какие из перечисленных ниже объявлений методов не являются допустимыми и по какой причине?

1. public final abstract int doAnything();
2. public abstract int doA(int j);
3. public private void notC(boolean flag);
4. final abstract boolean isEqual(float c, float d);

4. Контрольные вопросы

1. К какому виду исключительных ситуаций относится реализованная в упражнении 4.

2. Какие классы вы использовали для проверки типа элемента последовательности?

3. Почему все методы реализующие операции указанные в задании объявляются как статические?

4. Что такое расширяющее преобразование типов?

5.Что такое примитивный тип в Java?

Практическое занятие №2

Основы Языка Java, перегрузка и перекрытие методов, наследование

1. Цели работы

Целями работы являются:

- изучение принципов ООП в языке Java, использования перегрузки и перекрытия методов;
- получения представления о практическом назначении и использовании модификаторов объявлений классов, методов и полей
- получение навыков проектирования и реализации иерархии классов;
- изучение методов обработки исключительных ситуациях;
- получение навыков описания собственных исключительных ситуациях;

- получение навыков использования класс java.lang.Math для выполнения математических расчетов;

При решении задач и упражнений, вырабатываются навыки использования сужающего и расширяющего преобразования типов, создания объектов, использования логических конструкций, ветвлений и циклов.

Предметом изучения являются следующие классы и интерфейсы платформы J2SE:

- классы-оболочки над примитивными типами java.lang.Integer,java.lang.Double, java.lang.Float;
- класс объяляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;

2. Упражнения на программирование

1. Реализовать приложение, вычисляющее для последовательности чисел, представляющих величины углов, следующие тригонометрические функции:

- сумма синусов/косинусов значений,
- разность синусов/косинусов значений,
- произведение синусов/косинусов введенных значений.

Единица измерения углов должна указываться как параметр командной строки. Для представления единиц измерения и спользуйте константы с типом int. Для хранения значений, необходимо использовать массив с типом double. Объявления методов должны выглядеть следующим образом:

```
/**
@param angles – массив значений углов;
 @param type –единица измерения, имеющее значение одной из  констант
@return сумму синусов значений углов;
*/
public static double
          getSumSinuses(double[] angles,int type);
```

2. С помощью перегрузки реализуйте методы, позволяющие указывать количество значащих знаков для результатов выполнения операций.

3. Дополнить приложение, таким образом, чтобы имелась возможность формирования исходной числовой последовательности числами, сформированными генератором случайных чисел;

4. Реализовать класс объявляемой исключительной ситуации, для проверки корректности вводимых данных. Объект данной исключительной ситуации содержать в себе информацию о значении, обработка которого вызвала данную исключительную ситуацию.

5. Реализовать класс, в котором будут содержаться все описанные выше функциональные возможности.

3. Задачи

1. Пусть объявлен следующий класс:

```
package javaapplication2;
private class Sample {
      private static int value;
              static{
                     value=1;
                    }
     private int n;
          Sample(){
                      value++;
                    n=value%2;
                    }
     Sample(int n){
                     this();
                     n=this.n;
         }
      public  int getN(){
        return(n<10?n=value++:n);
    }
             }
```

Что произойдет в результате выполнения следующего метода?:

```
/*
```

```
    ..
**/
public void test(){
    for (int x=0;x<10;x++)
    {
    System.out.println(new Sample(x));
    }
    }
```

2. Какие из перечисленных объявлений полей класса являются недопустимыми и допустимыми и почему?

1. public final int 1v=2;
2. public final int 'xxx'=2;
3. public final int \u1000=1000;
4. public final static volatile boolean bool=true;
5. static volatile Boolean v2=Boolean.parseBoolean("true");

4. Контрольные вопросы

1. Дайте определения понятиям статический метод и статическое поле класса.
2. Что такое явное и неявное приведение типов?
3. Какие методы классов-оболочек над примитивными типами используются для получения значения примитивного типа из его строкового представления.
4. Для какого примитивного типа не существует класса-оболочки?

Практическое занятие № 3

Основы Языка Java. Перегрузка и перекрытие методов, наследование. Классы-оболочки

1. Цели работы

Целями работы являются:

- изучение принципов ООП в языке Java, использования перегрузки и перекрытия методов;
- получения представления о практическом назначении и использовании модификаторов объявлений классов, методов и полей

- получение навыков проектирования и реализации иерархии классов;
- изучение методов обработки исключительных ситуациях;
- получение навыков описания собственных исключительных ситуациях;
- получение навыков использования класс java.lang.Math для выполнения математических расчетов;
- получение навыков выбора оптимальной структуры библиотеки классов для решения поставленных задач;

При решении задач и упражнений, вырабатываются навыки использования сужающего и расширяющего преобразования типов, создания объектов, использования логических конструкций, ветвлений и циклов.

Предметом изучения являются следующие классы и интерфейсы платформы J2SE:

- классы-оболочки над примитивными типами java.lang.Integer, java.lang.Double, java.lang.Float;
- класс объявляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;

2. Упражнения на программирование

1. Реализовать приложение вычисляющее сумму, разность, произведение, деление n-степеней чисел введенной числовой последовательности для типов int, double, float, long.

2. Реализовать для приложения 1 возможность вывода результатов сравнения в системе счисления по выбору пользователя.

3. Реализовать библиотеку классов для вычисления функций описанных в упражнений. В качестве основы использовать интерфейс, прототип которого описывается следующим образом:

```
public interface Function{
    public long evaluate(int[] args);
    public double evaluate(double[] args);
    public double evaluate(float[] args);
```

```
        //...
    }
```

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего фрагмента исходного кода программы?

```
public class CustomerTwo {
    public static void main (String[] args) {
         Pizza favoritePizza = new Pizza();
         System.out.println("Meat on pizza before
                            baking: " +
                            favoritePizza.meat);
         bake(favoritePizza);
              System.out.println("Meat on pizza after
              baking: " + favoritePizza.meat);
         }
    public static void bake(Pizza pizzaToBeBaked) {
          pizzaToBeBaked.meat = "chicken";
          pizzaToBeBaked = null;
    }
}
class Pizza {
         String meat = "beef";}
```

4. Контрольные вопросы

1. Каким образом реализуется наследования в Java?
2. Каким образом класс-потомок может обратиться полям и методам суперкласса?
3. В какой последовательности осуществляется вызов конструкторов классов, являющихся суперклассами для данного класса?

Практическое занятие № 4
Основы Языка Java. Наследование, тригонометрические функции класса Math

1. Цели работы

Целями работы являются**:**

- изучение принципов ООП в языке Java, использования перегрузки и перекрытия методов;
- получения представления о практическом назначении и использовании модификаторов объявлений классов, методов и полей
- получение навыков проектирования и реализации иерархии классов;
- изучение методов обработки исключительных ситуациях;
- получение навыков описания собственных исключительных ситуациях;
- получение навыков использования класс java.lang.Math для выполнения математических расчетов;
- получение навыков выбора оптимальной структуры библиотеки классов для решения поставленных задач.

При решении задач и упражнений, вырабатываются навыки использования сужающего и расширяющего преобразования типов, создания объектов, использования логических конструкций, ветвлений и циклов.

Предметом изучения являются следующие классы и интерфейсы платформы J2SE:

- классы-оболочки над примитивными типами java.lang.Integer, java.lang.Double, java.lang.Float;
- класс объявляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;

3. Упражнения

1. Реализовать приложение, вычисляющее определенные интегралы для функций sin(x/n), cos(x/n), tan(x/n),cotan(x/n), для заданного диапазона в виде значений одного из следующих типов: int, double, float, long. Тип возвращаемого значения должен совпадать с типом передаваемых параметров.

2. Описать класс объявляемой исключительной ситуации TrigonometricException, генерируемой в случае возникновении ошибки при выполнении методов описанных в упражнении 1.

3. Реализовать класс, реализующий статические методы, предназначенные для выполнения операций из упражнения 1. Все методы должны быть объявлены как генерирующие исключительную ситуацию TriginometricException.

4. Реализовать библиотеку классов для вычисления функций описанных в упражнений 1. В качестве основы использовать интерфейс, прототип которого описывается следующим образом:

```
public interface Function{
public long evaluate(double lowMark, double
                        HighMark, int flow)
         throws TrigonometricException;
//...
}
```

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего фрагмента исходного кода?

```
class MultiArrays {
    public static void main(String[] args) {
        int[][] mXnArray = {
                    {16, 7, 12},
                    { 9, 20, 18},
                    {14, 11, 5},
                    { 8, 5, 10}
                    };
        int min = mXnArray[0][0];
```

```
            for (int i = 0; i < mXnArray.length; ++i)
                 for (int j = 0;j<mXnArray[i].length;
                      ++j)
                      min = Math.min(min,mXnArray[i][j]);
            System.out.println("Минимальное значение: "
                               + min);
        }
    }
```

4. Контрольные вопросы

1. Для чего предназначена секция импорта?
2. Какие существуют ограничения и правила именования классов?
3. Какие существуют способы инициализации массивов?

Практическое занятие № 5

Основы языка Java. Работа с изменяемыми и неизменяемыми строками

1. Цели работы

Целями практического занятия являются:

- получение навыков выполнения типовых операций над строками и буферами строк;
- изучение принципиальных отличий между классами String и StringBuffer;
- получение навыков принятия решения по оптимальному выбору между неизменяемыми и изменяемыми строками для решения поставленных задач;
- получение навыков обработки вводимых данных с консоли и представления их в требуемом виде;

Предметом изучения являются следующие классы и интерфейсы платформы J2SE

- класс для представления неизменяемых строк -java. lang.String;

- класс для представления изменяемых строк (строковых буферов) - java.lang.StringBuffer;

2. Упражнения на программирование

1. Требуется реализовать приложение, обладающее следующими функциональными возможностями: для заданного текста, поиск подстроки в строке, получение - количества слов, количества символов с учетом пробелов, без учета пробелов, количества гласных букв, количества согласных букв, поиск слов с максимальной и минимальной длиной. Для представления строк использовать класс java.lang.String и его методы;

2. Описать и реализовать класс, реализующий функционал, описанный в упражнении 1, с использованием статических методов класса. То есть, объявление класса, должно в целом соответствовать следующему виду:

```
public class StringUtils{
    public static char getFirstSymbolInString(String
                            searchString){
    return searchString.charAt(0);
    }
}
```

3. Дополнить класс, реализованный в упражнении 2, перегруженными методами, оперирующими объектами класса StringBuffer вместо объектов класса String;

4. Реализовать операцию поиска подстроки в строке на основе алгоритма QuickSearch (Быстрого поиска).

5. Реализовать на основе приложения, использующего класс реализованный в упражнении 3, возможность дополнения начального введеного текста.

3. Задачи

1. В приведенном ниже исходном тексте метода, определите сколько ссылок на объект создаваемый в строке (1) будет содержаться в момент выполнения строки (2).

```
public int getSomeValue(){
```

```
        Integer a=1; //(1)
        Integer b=new Integer(a);
        Integer c=a;
        Integer e=b;
        b=a;
        a=new Integer(100);
        return a+b+c+e; //(2)
    }
```

2. Какая из строк приведенного ниже метода может выбросить исключительную ситуацию NumberFormatException?

```
public void parseNumbersFromString(String numb){
    java.util.List<Integer> numbers=
        new java.util.LinkedList<Integer>();
    java.util.List<Integer> separators=
        new java.util.LinkedList<Integer>();
    int i=0;
    while ((i!=-1)&&(i<numb.length()))
    {
        int indsep=numb.indexOf(',', i);
        if (indsep!=-1)
        separators.add(indsep);
        i=indsep+1;
    }
    separators.add(0, 0);
    separators.add(numb.length());
    Iterator it=separators.iterator();
    while (it.hasNext())
    {
        numbers.add(new Integer(numb.substring(
         ((Integer)it.next()).intValue()+1,
         ((Integer)it.next())-1)));
    }
}
```

4. Контрольные вопросы

1. В чем состоит принципиальное отличие между классами String и StringBuffer?

2. Какие типы исключительных ситуаций вы знаете? К исключительным ситуациям какого типа относится java.lang.Error?;
3. Допустимо ли преобразование объекта класса Integer к типу String?
4. Какие методы предусмотрены в классе Integer для представления числовых значений в различных системах счисления?

Практическое занятие № 6

Основы языка Java. Наследование. Сравнение объектов

1. Цели работы

Целями практического занятия являются:

- изучение механизмов сравнения объектов в языке Java с использованием интерфейсов Comparable и Comparator;
- получение навыков работы с файловыми потоками ввода-вывода;
- получение навыков реализации библиотек классов со сложной структурой;
- получение навыков работы с классами-оболочками над простыми типами данных;

Предметом изучения являются следующие классы и интерфейсы платформы J2SE

- классы-оболочки над примитивными типами java.lang.Integer,java.lang.Double, java.lang.Float;
- класс объяляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;
- интерфейс java.lang.Compare;
- интерфейс java.lang.Comparable;
- классы пакета поддержки ввода-вывода -java.io.IOException, java.io.FileWriter;

2. Упражнения на программирование

1. Реализовать библиотеку классов, представляющих геометрические фигуры - окружность, ромб, параллелограмм, трапецию, треугольник. Реализовать методы вычисления площади, периметра, длины ребер, диаметров вписанной и описанной окружности. В качестве основы библиотеки описать интерфейс или абстрактный класс Figure, являющийся абстракции геометрической фигуры. Необходимо реализовать следующие методы -вычисление площади, периметра, высоты, длины ребер(для многоугольников), радиуса, диаметра (для окружности), медиан, биссектрис. Для указания вершин использовать координаты декартовой системы координат. Предусмотреть конструкторы с параметрами.

2. Расширить иерархию классов из упражнения 1, дополнив ее некоторыми частными случаями (прямоугольник, квадрат, равносторонний, равнобедренный и прямоугольный треугольник).

3. Реализовать наследник класса Comparator и реализовать интерфейс Comparable для классов описанных в упражнении 1 и 2, для получения возможности сравнения геометрических фигур по величине их площади.

4. Реализовать функцию записи содержимого объектов в файл.

3. Задачи

1. Что произойдет в результате выполнения следующего кода:

public class Sample {

```
public static void main(String[] args) {
    String[][][] arr ={
                       { {}, null },
                       { { "1", "2" }, { "1", null,
                       "3" } },
                       {},
                       { { "1", null } }
                       };
    System.out.println(arr.length +
                                arr[1][2].length);
    }
}
```

2. Какие из нижеперечисленных объявлений полей класса являются допустимыми?

а) int morrow=1;

б) public transient static x=new String();

в) java.lang.Integer.MAX_INTEGER val=new Integer(100);

г) java.math.BigDecimal dbm=new java.math.BigDecimal();

4. Контрольные вопросы

1. Для чего предназначена перегрузка методов класса?
2. Могут ли перекрываться статические методы класса в классах потомках?
3. Какие основные задачи решает класс File?
4. Для чего предназначен интерфейс Map?

Практическое занятие № 7
Основы языка Java. Наследование. Сравнение объектов. Запись в файловый поток

1. Цели работы

Целями практического занятия являются:

- получение навыков проектирования и разработки библиотеки классов;
- получение навыков реализации механизмов сравнения объектов в соответствии с заданным порядком;
- изучение основных классов и интерфейсов, а также их методов и свойств, для организации файлового потока ввода-вывода;

В ходе выполнения практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы-оболочки над примитивными типами java.lang.Integer,java.lang.Double, java.lang.Float;
- класс объяляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;

- интерфейс java.lang.Compare;
- интерфейс java.lang.Comparable;
- классы пакета java.io – IOException, FileWriter, FileReader, FileInputStream, BufferedReader;

2. Упражнения на программирование

1. Реализовать библиотеку классов представляющих собой абстракцию организационной структуры предприятия. Разработать следующие классы – «Человек», «Сотрудник», «Подразделение», «Должность», соответствующие следующим требованиям

- Класс «Человек» должен обладать как минимум следующими свойствами - фамилия, имя, отчество, дата рождения, пол.
- Класс «Сотрудник» должен расширять класс «Человек» обладать как минимум следующими свойствами – подразделение, должность, зарплата. Для сотрудника, должны быть доступны прием и увольнение на работу.
- Классы подразделения должны иметь иерархическую структуру, то есть каждое подразделение может иметь главное и подчиненные подразделение. Должны быть доступны ввод/вывод подразделения организации.

5. Расширить иерархию классов из упражнения 1, дополнив ее иерархией должностей. Ввести понятие штатной должности. В отличии от должности, штатная должность связанна с конкретным подразделением.

6. Реализовать наследник класса Comparator и реализовать интерфейс Comparable для классов, описанных в упражнении 1 и 2, для получения возможности сравнения по уровню подчиненности.

7. Реализовать методы вычисления численности сотрудников, величины средней заработной платы, суммарной заработной платы.

8. Реализовать функцию записи содержимого объектов в файл.

3. Задачи

1. Что произойдет в результате выполнения следующего программного кода?

```
public class Sample {
     static int a;
     int b;
     public Q275d() {
     int c;
     c = a;
     a++;
     b += c;
}
public static void main(String[] args) {
                    new Sample();}
 }
```

2. Какие из приведенных ниже объявлений классов являются правильными?

1. public static class Sample{...};
2. interface LowInterface{...}; protected class Impl extends LowInterface{...};
3. abstract class A {...}
4. final abstract class B{...};
5. final class X{...};

4. Контрольные вопросы

1. Можно ли производить запись в файл с использованием класса OutputStreamWriter?
2. Для чего предназначены буферизованные потоки чтения и записи?
3. Что понимается под блоком инициализации класса?
4. Каким образом функционирует «сборщик мусора» виртуальной машины Java?

Практическое занятие № 8

ООП в JAVA, наследование, сериализация, файловые потоки ввода-вывода

1. Цели работы

Целями практического занятия являются:

- получение навыков проектирования и разработки библиотек классов;
- изучение и получение навыков использования механизма сериализации объектов;

В ходе работы изучаются следующие классы и интерфейсы платформы J2SE

- классы пакета java.lang – String, StringBuffer, Double,Long,Integer;
- класс объяляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;
- интерфейс java.io.Serializiable;
- классы реализующие потоки чтения/записи объектов -java.io.ObjectInputStream,java.io.ObjectOutputStream;
- классы, реализующие файловые потоки ввода-вывода -java.io.FileReader, java.io.FileWriter, java.io.FileInputStream.

2. Упражнения на программирование

1. Спроектировать и реализовать классы, эмулирующие базу данных для операций над банковскими счетами – банковский счет, владелец счета, история операций.

Должны быть доступны следующие операции

- депонирование некоторой суммы на счет;
- снятие некоторой суммы со счета;
- передача суммы с одного счета на другой;
- открытие счета и закрытие счета;
- получение истории операций;
- регистрация клиента;
- получение баланса по счету;
- формирование сводной информации по всем счетам;

2. Модифицировать библиотеку классов из упражнения 1, реализовав иерархию классов банковских операций.

3. Реализовать интерфейс Serializiable в классах реализованных в упражнениях 1 и 2 для сохранения и восстановления объектов в файл/из файла.

4. Реализовать класс управляющий механизмами сохранения и восстановления объектов системы, объект которого будет статической переменной приложения.

3. Задачи

1. В каком из приведенных объявлений методов содержится ошибка:

а) public final someMethod(int x; final long a);

б) public abstract final someAnotherMethod(int x; long b);

в) public synchronized void sMethod();

2. Что произойдет в результате выполнения следующего кода:

```
public class Sample {
        static void test(int i) {
             int j = i/2;
             int k = i >>> 1;
             assert j == k : i;
        }
        public static void main(String[] args) {
         test(0);
         test(2);
         test(-2);
         test(1001);
         test(-1001);
        }
    }
```

4. Контрольные вопросы

1. Для чего предназначен интерфейс Serializiable?
2. Какие виды наследования поддерживаются в языке Java?
3. Для чего предназначены итераторы?
4. Какие существуют способы задания констант в Java?

Практическое занятие № 9

ООП в JAVA, наследование, сериализация, файловые потоки ввода-вывода

1. Цели работы

Целями практического занятия являются:

- получение навыков проектирования и разработки библиотек классов;
- изучение и получение навыков использования механизма сериализации объектов;

В ходе работы изучаются следующие классы и интерфейсы платформы J2SE

- классы пакета java.lang – String,StringBuffer, Double, Long, Integer;
- класс объявляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;
- интерфейс java.io.Serializiable;
- классы реализующие потоки чтения/записи объектов -java.io.ObjectInputStream, java.io.ObjectOutputStream;
- классы, реализующие файловые потоки ввода-вывода -java.io.FileReader, java.io.FileWriter, java.io.FileInputStream.

2. Упражнения на программирование

1. Реализовать библиотеку классов, для эмуляции базы данных склада. Реализовать классы «Склад», «Товар», «Покупатель», «Поставщик».

Должны быть реализованы следующие функции:

- добавление единицы товара;
- отпуск товара покупателю;
- получение информации о доступных позициях товара;

- получение информации о наличии товара на складе;
- группировка товаров по видам;
- получение информации по поставщикам;
- получение информации по покупателям;

2. Расширить библиотеку классов из упражнения 1, реализовав журнал выполнения операций с товарами.

3. Реализовать интерфейс Serializiable в классах реализованных в упражнениях 1 и 2 для сохранения и восстановления объектов в файл/из файла

4. Реализовать класс управляющий механизмами сохранения и восстановления объектов системы, объект которого, будет статической переменной приложения.

3. Задачи

1. Что произойдет в результате выполнения следующего кода программы:

```
class MyException extends Exception {}
        public class Sample {
            public void foo() {
                    try {
                        bar();
                    } finally {
                        baz();
                    } catch (MyException e) {}
            }
        public void bar() throws MyException {
                throw new MyException();
        }
        public void baz() throws RuntimeException {
            throw new RuntimeException();
                    }
    }
```

4.Контрольные вопросы

1. Какие классы в Java предназначены для работы с файлами?
2. Что подразумевает сериализация объектов?

3. Какие существуют средства поддержки механизма сериализаии в Java?
4. Для чего предназначен класс BufferedReader?

Практическое занятие № 10

ООП в Java. Наследование

1. Цели работы

Целями работы являются**:**

- изучение принципов ООП в языке Java, использования перегрузки и перекрытия методов;
- получения представления о практическом назначении и использовании модификаторов объявлений классов, методов и полей
- получение навыков проектирования и реализации иерархии классов;
- изучение методов обработки исключительных ситуациях;
- получение навыков описания собственных исключительных ситуациях;
- получение навыков использования класс java.lang.Math для выполнения математических расчетов;
- получение навыков выбора оптимальной структуры библиотеки классов для решения поставленных задач.

При решении задач и упражнений, вырабатываются навыки использования сужающего и расширяющего преобразования типов, создания объектов, использования логических конструкций, ветвлений и циклов.

Предметом изучения являются следующие классы и интерфейсы платформы J2SE:

- классы-оболочки над примитивными типами java.lang.Integer, java.lang.Double, java.lang.Float;
- класс объявляемой исключительной ситуации java.lang.Exception;
- класс java.lang.Math;

2. Упражнения на программирование

1. Реализовать модель следующей системы- есть некоторый прибор для измерения температуры(термометр) и некоторое количество приборов регулирующих температуру, путем оказания на них некоторого воздействия. Система в начальный момент времени имеет некоторую начальную температуру, задаваемую при инициализации, а также значением предельно допустимой температуры . Каждый прибор характеризуется максимально возможной температурой, удаленностью от термометра, шагом нагрева, шагом охлаждения, временем реакции скоростью нагрева, скоростью охлаждения, дальностью воздействия. Также известно, что области воздействия не перекрываются. Термометры находятся на некотором удалении от приборов регулирующих температуру, и с некоторым заданным интервалом уведомляют систему о текущей температуре. При достижении максимальной температуры приборам дается уведомление о снижении температуры. Реализовать классы «термометр», «кондиционер», «централизованный контроллер»

2. Реализовать приложение, использующее классы, описанные в упражнении 1 и позволяющее управлять системой и выводить информацию о ее состоянии.

3. Реализовать потомок класс «кондиционер» обладающий возможностью автоматически останавливать процесс нагрева и охлаждения.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода программы:

```
public class TestClass{
private static double a ;
     static{
     a=System.getCurrentTime();
     a=a++;
```

```
        }
        public long b;
        TestClass{
        b=a;
        }
}
```

2. Какое из приведенных объявлений констант не является верным:

1. public final static int k=0;
2. public final static long x=System.getCurrentTime();
3. public final static long c=100000l;
4. public final static volatile int z=2;

4. Контрольные вопросы

1. Какие основные методы класса String вам известны?
2. Какие существуют правила перекрытия методов в Java?
3. Каким образом используются диагностические утверждения?

Практическое занятие № 11

Прикладные классы платформы J2SE, коллекции

1. Цели работы

Целями практического занятия являются

- изучение возможностей платформы J2SE для решения прикладных задач;
- методы получения и обработки значений вида дата/время;
- получение навыков применения региональных настроек и локализации приложений;
- получение навыков выполнения операций по чтению/записи в файловые потоки ввода/вывода;
- применение нестандартного порядка для организации множеств;

Предметом изучения являются следующие классы и интерфейсы платформы J2SE

- java.util.Date, java.util.Calendar;
- java.lang.System;

- java.util.SortedSet, java.util.Set;
- java.io.FileInputStream, java.io.FileOutputStream, java.io.FileReader, java.io.BufferedReader, java.io.FileWriter;

3. Упражнения на программирование

1. Реализовать приложение выводящее значение текущей даты и времени с определенным интервалом с выбранными параметрами локализации;

2. Реализовать класс, позволяющий выполнять функции описанные в упражнении 1 и сохранять результаты в файл;

3. Реализовать класс позволяющий сохранять результаты работы приложения из упражнения 1 в объект класса SortedSet и упорядочивать их в следующем порядке -миллисекунды, секунды, часы, дни, месяцы, годы.

4. Дополнить приложение из упражнения 2 возможностью вывода значений часа, минут, секунд, дня, месяца, года в различных системах счисления.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
public class Average8 {
    public static void main(String[] args) {
        try { // (1)
        printAverage(100, 0); // (2)
    } catch (IntegerDivisionByZero idbze) { // (3)
    idbze.printStackTrace();
    System.out.println("Исключение отловлено в: »+
                       "main().");
    } finally { // (4)
    System.out.println("Finally выполнено в
                       main().");
    }
    System.out.println("Exit main()."); // (5)
}
```

```
public static void printAverage(int totalSum, int
                                 totalNumber)
            throws IntegerDivisionByZero { // (6)
        int average = computeAverage(totalSum,
                                     totalNumber);
        System.out.println("Average = " +
        totalSum + " / " + totalNumber + " = " +
        average);
System.out.println("Выход printAverage()."); // (7)
}

public static int computeAverage(int sum,
                                 int number)
throws IntegerDivisionByZero { // (8)

System.out.println("Вычисление среднего.");
if (number == 0) // (9)
throw new IntegerDivisionByZero("Деление на ноль");
return sum/number; // (10)
}
}
```

4. Контрольные вопросы

1. Каким образом в Java можно получить значение текущей времени и даты?
2. Для чего предназначен класс Calendar?
3. Какой класс позволяет получить текущую дату в соответствии с региональными параметрами?

Практическое занятие № 12

Обработка изменяемых строк, коллекции, карты

1. Цели работы

Целями работы являются:

- получение навыков выполнения типовых операций над строками и строковыми буферами;
- получение навыков разработки собственных классов, реализующих стандартные интерфейсы платформы J2SE;

- изучение методов организации сравнения объектов с использованием нестандартных правил упорядочивания;
- получение навыков анализа производительности приложений в рамках платформы J2SE;

В лабораторной работе предусматривается изучение следующих классов и интерфейсов платформы J2SE**:**

- java.lang.String, java.lang.StringBuffer;
- java.util.Comparator, java.util.Comparable;
- java.util.Map, java.util.Hashmap;
- java.util.Timer;

3. Упражнения на программирование

1. Реализовать интерфейс Comparator, позволяющий сравнить объекты классов String и StringBuffer по длине, без учета пробелов.

2. Реализовать интерфейс Comparator, позволяющий сравнить объекты классов String и StringBuffer, в обратном порядке следования символов.

3. Реализовать собственную реализацию интерфейса Map.

4. Произвести сравнительный анализ скорости выполнения операций при использовании объектов собственной реализации класса Map и класса Hashmap. Результаты сравнения описать в виде таблицы, приведенной ниже

Операция	Количество объектов	java.util.Hashmap	Собствен. реализация
Добавление элемента	1000		
	5000		
	6000		
	10000		
	15000		
	20000		

	...		
Удаление элемента	1000		
	5000		
	...		
Получение элемента	...		
Поиск элемента	...		

3. Задачи

1. Что будет выведено на экран в результате компиляции и выполнения следующего кода:

```
class A{}
class B extends A{}
class C extends A{}
class D extends B{}
class Main{
    public static void main (String[] args}{
         A a=new D();
         B b=new B();
         D d=new D();
         C c=new A();
         System.out.println (( a instanceof A)||(d
              instance of A)||
                   c instanceof A)||(d instanceof C));
    }
}
```

2. Что будет выведено на экран в результате выполнения следующего программного кода:

```
public class Q8499 {
    public static void main(String[] args) {
            double d = -2.9;
            int i = (int) d;
            i *= (int) Math.ceil(d);
```

```
                i *= (int) Math.abs(d);
                System.out.println(i);
                                    }
        }
```

4. Контрольные вопросы

1. Какой метод организации данных использует класс Hashmap?

2. Опишите иерархию классов и интерфейсов коллекций в платформе J2EE?

3. Для чего предназначены цепочки конструкторов?

4. Какие методы существуют для получения значения системного времени в J2SE?

Практическое занятие № 13

Коллекции и списки

1. Цели работы

Целями практического занятия являются:

- изучение методов и свойств базовых интерфейсов и классов представляющих коллекции, списки и карты;
- получение навыков реализации стандартных интерфейсов платформы J2SE;
- получение навыков использования вложенных интерфейсов и классов;
- получение навыков выполнения операций со стандартными потоками ввода-вывода;
- изучение методов сравнения объектов;

В ходе практического занятия изучаются методы и свойства следующих классов и интерфейсов платформы J2SE

- классы и интерфейсы для работы с коллекциями и картами -java.util.Collection, java.util.Set,java.util.SortedSet, java.util.List, java.util.LinkedList, java.util.Map, java.util.Hashmap;

- интерфейсы предназначенные для реализации механизма сравнения объектов -java.util.Comparator, java.util.Comparable;
- классы предназначенные для выполнения операций с потоками ввода-ввывода -java.io.OutputStream, java.io.OutputStreamWriter, java.io.IOException;
- java.lang.String;
- java.lang.System;

2. Упражнения на программирование

1. Реализовать интерфейс ListsComparator для объектов классов LinkedList, сравнивающих 2 объекта по количеству элементов содержащихся в списке.

2. Реализовать приложение, принимающее массивы строк и формирующее объекты класса LinkedList, содержащее слова строки, и помещающее полученные объекты в объект класса SortedSet, созданный с использованием реализованного в упражнении 1 объекта класса Comparator и выводящее на экран содержимого созданного объекта класса SortedSet.

3. Реализовать класс выполняющий:

- поиск всех вхождений подстроки в строке,
- всех вхождений заданной подстроки в списке строк (объектов классов реализующих интерфейс List),
- всех элементов массива строк совпадающих с искомой подстрокой,
- всех вхождений искомой строки в объект реализующий интерфейс Map.

Результат должен возвращаться в виде объекта вложенного в класс локального класса, и обладающего следующими полями – время проведения поиска, длительность поиска, индексы начала вхождений при поиске подстроки в строке, индексы элементов при поиске в списке и ключи при поиске в картах.

4.Описать интерфейс Printable, позволяющий выводить в стандартный поток содержимое некоторого объекта. Породить класс на-

следующий от класса реализованного в упражнении в 3 для представления результатов поиска и реализующего интерфейс Printable, для вывода результаты поиска в стандартный поток.

3. Задачи

1. Верно ли следующее объявление интерфейса:

```
public interface A{
    public final static int A=10;
    public abstract void methodB();
    public int method C();
    public static int methodA(){
    return A;
}
}
```

2. Что произойдет при компиляции и выполнении следующего исходного кода:

```
public class Sample {
        int a;
        int b;
        public void f() {
            a = 0;
            b = 0;
            int[] c = { 0 };
            g(b, c);
            System.out.println(a + " " + b +
            " " + c[0] + " ");
        }
        public void g(int b, int[] c) {
        a = 1;
        b = 1;
        c[0] = 1;
        }
    public static void main(String[] args) {
    Sample obj = new Sample();
    obj.f();
    }
}
```

4. Контрольные вопросы

1. Для чего предназначены интерфейсы Comparator и Comparable? Реализует ли интерфейс Comparable класс Boolean?
2. Для чего предназначены неизменяемые оболочки коллекций?
3. Какие существуют способы инициализации массивов?
4. Для чего предназначен интерфейс Map?

Практическое занятие № 14

Прикладные классы J2SE. Класс Bitset

1. Цели работы

Целями работы являются:

- получение навыков использования прикладных классов J2SE на примере класса, представляющего абстракцию последовательности битов;
- получение навыков выполнения операций со стандартными потоками ввода-вывода;

В ходе практического занятия изучаются следующие классы и интерфейсы:

- класс для выполнения операций над последовательностью битов – java.util.Bitset;
- классы представляющие средства для оперирования со стандартными потоками ввода-вывода -java. io.OutputStream, java.io.OutputStreamWriter;

2. Упражнения на программирование

1. Реализовать класс IntegerBitset, предназначенный для представления числового значения типа Integer в виде четырехбайтового значения, каждый байт- последовательность 8 битов хранящихся в виде значений типа Bitset. Класс IntegerBitset должен содержать методы по преобразованию значений из типа Integer, массива объектов класса Bitset и строкового представления данных значений.

2. Реализовать класс, представляющий собой матрицу произвольных размеров элементов класса IntegerBitset и реализующий стандартные операции над матрицами, при чем все операции должны выполняться над последовательнастями битов.

3. Описать интерфейс Printable, позволяющий выводить в стандартный поток содержимое некоторого объекта. Породить класс наследующий от класса реализованного в упражнении в 2 и реализующий интерфейс Printable, для вывода содержимого объекта в стандартный поток вывода.

3. Задачи

1. Что произойдет при попытке компиляции и запуска следующего фрагмента?

```
public class Q275d {
        static int a;
        int b;
        public Q275d() {
             int c;
             c = a;
             a++;
             b += c;
        }
    public static void main(String[] args) {
    new Q275d();
    }
}
```

2. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
public abstract class Main {
        class TestOutput{
             LinkedList numbers=new LinkedList();
             TestOutput(long count )
             {
                  for (long i=0;i<count;i++){
                       numbers.add(Math.random());
             }
```

```
        }
        public void addNumbers(long count){
            for (long i=0;i<count;i++)
            numbers.add(Math.random());
        }
    public void writeToFile(String fileToWrite)
         throws IOException
    {
        java.io.FileWriter fw=
            new java.io.FileWriter(fileToWrite);
        try{
            Iterator i=numbers.iterator();
            while (i.hasNext()){
                Double d=(Double)i.next();
                fw.write(d.toString());
                fw.close();
            i.remove();
            }
        }catch (IOException e){
            throw e;
        }
        finally{
            fw.close();
        }
        // fw.write();
    }
}
public static void main(String[] args) {
// TODO code application logic here
    try{
        (new TestOutput(Long.parseLong
                (args[1]))).writeToFile(args[0]);
    }
    catch (IOException e)
    {
    System.err.print(e);
    }
}
```

```
    }
}
```

4. Контрольные вопросы

1. Какие виды вложенных классов и интерфейсов вы знаете?
2. В чем заключаются принципиальные различия между абстрактными классами и интерфейсами?
3. Какие модификаторы доступа допустимы для методов класса?

Практическое занятие № 15

Наследование. Стандартные потоки ввода-вывода

1. Цели работы

Целями практического занятия являются -

- изучение принципов объектного-программирования и их реализации в языке Java;
- получение навыков выполнения операций со стандартными потоками ввода-вывода;
- получение навыков работы с классами-оболочками над примитивными типами;
- получение навыков обработки исключительных ситуаций и реализации собственных классов исключительных ситуаций;

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы и интерфейсы представляющие коллекции и списки- java.util.List, java.util.LinkedList;
- классы оболочки над примитивными типами – java.lang.Integer, java.lang.Double, java.lang.Long, java.lang.Float;
- классы для работы со стандартным потоком вывода -java. io.OutputStream, java.io.OutputStreamWriter, java.io.IO.IOException;

2. Упражнения на программирование

1. Реализовать класс являющийся абстракцией матрицы объектов класса Element. Все элементы класса матрицы должны иметь одинаковый тип, в противном случае необходимо генерировать исключительную ситуацию. Должны быть реализованы следующие методы:

- сложение матриц;
- вычитание матриц;
- умножение матриц;
- подсчет суммы элементов;

2. Реализовать иерархию классов наследующих от Element – целочисленные элементы, элементы с плавающей точкой.

3. Описать интерфейс Printable, позволяющий выводить в стандартный поток содержимое некоторого объекта. Породить класс наследующий от класса реализованного в упражнении в 2 и реализующий интерфейс Printable, для вывода содержимого объекта в стандартный поток вывода.

4. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
class NonStaticForwardReferences {
{
    nsf1 = 10;
    nsf1 = sf1;
    int a = 2 * nsf1;
    int b = nsf1 = 20;
    int c = this.nsf1;
}
    int nsf1 = nsf2 = 30;
    int nsf2;
    static int sf1 = 5;
{
    int d = 2 * nsf1;
    int e = nsf1 = 50;
}
public static void main(String[] args) {
```

```
        NonStaticForwardReferences objRef =
                 new NonStaticForwardReferences();
        System.out.println("nsf1: " + objRef.nsf1);
        System.out.println("nsf2: " + objRef.nsf2);
        }
    }
```

2. Метод какого класса будет вызван в результате выполнения следующего исходного кода:

```
public class A{};
public class B extends A{
         public void doSomething(){...}};
         public class C extends B{
             public void doSomething(){...}};
         }
         public class Test{
              public void main (String[] s)
              {
                   C c=new C();
                   A a=(A)(B)c;
                   a.doSomething();
              }
         }
```

4. Контрольные вопросы

1. Какой класс удобнее всего использовать для хранения пар типа «ключ-значение»?
2. Существуют ли методы позволяющие сохранять текст в файл с выбранной кодировкой?
3. Какие кодировки поддерживаются для объектов класс String?

Практическое занятие № 16

Наследование. Стандартные потоки ввода-вывода

1. Цели работы

Целями практического занятия являются -

- изучение принципов объектно-ориентированного программирования и их реализации в языке Java;

- получение навыков работы с буферизованными потоками ввода-вывода;
- получение навыков выполнения операций со стандартными потоками ввода-вывода;
- получение навыков работы с классами-оболочками над примитивными типами;
- получение навыков обработки исключительных ситуаций и реализации собственных классов исключительных ситуаций;

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы и интерфейсы представляющие коллекции и списки- java.util.List,java.util.LinkedList;
- классы оболочки над примитивными типами – java.lang.Integer, , java.lang.Long;
- классы для работы со стандартными потоком вывода -java. io.BufferedOutputStream, java.io.OutputStreamWriter, java.io.IOException, java.io.BufferedReader, java.BufferedWriter;

2. Упражнения на программирование

1. Реализовать библиотеку классов для операций над матрицами целых чисел. Для представления строк и столбцов матрицы должен использоваться класс LinkedList.

Должны быть реализованы следующие операции:

- изменение элемента;
- обнуление элемента;
- изменение размеров матрицы;
- вычисление суммы элементов матрицы;
- вычисление произведения суммы матриц;
- сложение текущего объекта класса матрицы с другим, с созданием нового объекта класса «Матрица»
- вычитание из одной матрицы другой;
- умножение двух матриц;

2. Реализовать приложение, которое обеспечивает заполнение объекта класса, реализованного в упражнении 1, из стандартного потока чтения, побайтовым чтением и занесением считанных данных в промежуточный буфер и побайтовый вывод содержимого объекта в стандартный поток вывода.

3. Реализовать заполнение объекта из байтового потока чтения, считывающего только неотрицательные значения. Отрицательные значения должны быть снова занесены в поток.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
class SuperclassA {
    public SuperclassA() {
       System.out.println("Конструктор SuperclassA");
    }
}
class SubclassB extends SuperclassA {
    SubclassB() {
        this(3);
    System.out.println("Конструктор
                          по-умолчанию SubclassB");
    }
    SubclassB(int i) {
        System.out.println(«Перегруженный
                          конструктор в SubclassB");
        value = i;
    }
    {
        System.out.println("Блок инициализации
                               SubclassB");
        value = 2;
    }
    int value = initializerExpression();
    private int initializerExpression() {
    System.out.println("Инициализатор поля
                     класса в SubclassB");
```

```
        return 1;
    }
    }
    public class ObjectConstruction {
        public static void main(String[] args) {
            SubclassB objRef = new SubclassB(); // (8)
            System.out.println("value: "+ objRef.value);
        }
    }
```

2. Какое из объявлений переменных содержит ошибку:
    1. private transient int a=100;
    2. private statict volatile int v=1000;
    3. final public String l= «aaaa»+ «bbbb»;

4. Контрольные вопросы

1.Что подразумевает понятие рефлексии в Java?

2. Для чего предназначен модификатор transient?

3.Для каких целей предназначен метод sleep у потока?

Практическое занятие № 17

Файлы. Файловые потоки ввода-вывода

1. Цель работы

Целями практического занятия являются:

1. получение навыков выполнения операций с файлами средствами платформы J2SE;
2. получение навыков обработки параметров командной строки;

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы для выполнения операций с файлами – java.io.File, java.io.FileInputStream, java.io.FileOutputStream, java.io.FileReader, java.io.FileWriter,java.io.IOException, java.io.BufferedReader;
- класс java.lang.Math;
- класс java.lang.System;

2. Упражнения на программирование

1. Реализовать приложение, заносящие в файл последовательность случайных чисел в диапазоне, указываемом с помощью параметров командной строки. Имя файла указывается также в командной строке.

2. Дополнить приложение, реализованное упражнении 1, функцией поиска числа, заданного в командной строке, в файле, в который ранее была записана числовая последовательность.

3. Дополнить приложение реализованное в упражнении 1 и 2, возможностью дозаписи новых элементов последовательности, с контролем неповторяемости значений. Новые числа могут быть как сгенерированы случайным образом, так и введены с консоли.

4. Реализовать вывод содержимого файла сгенерированного в упражнениях 1-3, в консоль построчно, с поддержкой функций управления - переход к следующей строке и предыдущей.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
public class Q28fd {
    public static void main(String[] args) {
         int counter = 0;
         l1:
         for (int i=0; i<10; i++) {
         l2:
         int j = 0;
         while (j++ < 10) {
              if (j > i) break l2;
                   if (j == i) {
                        counter++;
                        continue l1;
              }
    }
}
    System.out.println(counter);
}
```

```
}
```

2. Правильно ли следующее объявление класса?

```
public abstract class N implements C{
    private long a;
    final abstract void doNothing();
}
```

4. Контрольные вопросы

1. Какие существуют классы расширяющие класс InputStream и каково их назначение?

2. Какие классы предназначены для преобразования из байтового потока вывода в символьный поток вывода?

3. Каким образом можно получить данные о размере файла?

Практическое занятие № 18

Файлы, операции с файлами

1. Цели работы

Целями практического занятия являются:

- получение навыков выполнения операций с файлами (создание,чтение, запись, поиск в файле) средствами платформы J2SE;
- работа с атрибутами файлов;
- получение навыков обработки параметров командной строки;

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы для выполнения операций с файлами – java.io.File, java.io.FileInputStream, java.io.FileOutputStream, java.io.FileReader, java.io.FileWriter,java.io.IOException, java.io.BufferedReader;
- класс java.lang.Math;
- класс java.lang.System;

2. Упражнения на программирование

1. Реализовать приложение, которое объединяет содержимое нескольких текстовых файлов, указанных в командной строке, создает новый файл в который записывает объединенное содержимое и помечает его как доступный только для чтения.

2. Реализовать приложение, которое разбивает содержимое текстового файла на несколько частей в соответствии с числом, указанным в параметре командной строки и сохраняет каждую часть во вновь создаваемый файл. При этом исходный файл удаляется, а для каждого вновь созданного устанавливается атрибут «только для чтения».

3. Реализовать поиск в файле заданной подстроки и ее удаление в случае, если она была обнаружена. При этом в консоль должно выводится значение смещения в символьном эквиваленте начала очередного вхождения.

4. Реализовать приложение, производящее поиск всех файлов в директории и ее поддиректориях с выводом информации об этих файлах;

5. Реализовать приложение, производящее поиск всех файлов в директории и ее поддиректориях с выводом информации об этих файлах, удовлетворяющих следующим критериям:

- дата создания;
- дата последнего изменения;
- установленные атрибуты;
- соответствие заданной маске имени файла;

6. Реализовать поиск подстроки в тексте файлов, найденных в заданной директории.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
class Base {
    int i;
    Base() { add(1); }
```

```
    void add(int v) { i += v; }
    void print() { System.out.println(i); }
}
class Extension extends Base {
    Extension() { add(2); }
    void add(int v) { i += v*2; }
}
public class Qd073 {
    public static void main(String[] args) {
    bogo(new Extension());
}
static void bogo(Base b) {
    b.add(8);
    b.print();
    }
}
```

2. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
public class Q03e4 {
    public static void main(String[] args) {
    String space = " ";
    String composite = space + "hello" +
                  space + space;
    composite.concat("world");
    String trimmed = composite.trim();
    System.out.println(trimmed.length());
    }
}
```

4. Контрольные вопросы

1. На основе какой структуры данных реализован класс TreeSet?

2. Для каких целей используется модификатор final в объявлении классов?

3. Какова область видимости у классов без явно указанного модификатора доступа?

Практическое занятие № 20

Многопоточные приложения

1. Цели работы

Целями практического занятия являются:

- изучение принципов реализации многопоточных приложений;
- получение навыков разработки многопоточных приложений;
- получение синхронизации операций над разделяемыми объектами в условиях многопоточности;

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы и интерфейсы пакета java.lang.* -Runnable, Thread;
- класс объяляемой исключительной ситуации java.lang.Exception;
- классы пакета java.io.* -FileReader, FileWriter, IOException, BufferedReader, FileInputStream,FileOutputStream;

2. Упражнения на программирование

1. Реализовать многопоточное приложение, каждый поток которого осуществляет поиск файлов в указанной директории, при этом список директорий формирует главный поток приложений.

2. Модифицировать приложение реализованное в упражнении 1, следующим образом. Пусть, приложению, в качестве параметра входной строки передается некоторая директория для поиска файлов. После запуска поиска при нахождении каждой директории запускается новый поток для поиска внутри нее, если не существует свободных потоков. Если существует свободный поток то он активизируется для выполнения операции поиска. По завершению поиска поток засыпает.

3. Дополнить приложение 2 выводом статистической информации о работе приложений, включающей в себя следующую – измене-

ние числа потоков, время активной работы потоков, количество найденных файлов, процентное соотношение времени выполнения между потоками.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
public class Vertical {
    private int alt;
    public synchronized void up() {
        ++alt;
    }
    public void down() {
        --alt;
    }
    public synchronized void jump() {
        int a = alt;
        up();
        down();
        assert(a == alt);
    }
}
```

4. Контрольные вопросы

1. Какая последовательность вызова методов классов пакета java.io. при осуществлении чтения из файлового потока?
2. Объясните понятие и назначение финализации объектов.
3. Приведите примеры использования класса ArrayList.

Практическое занятие № 19

Многопоточные приложения. Синхронизация

1. Цели работы

Целями практического занятия являются:

- изучение принципов реализации многопоточных приложений;
- получение навыков разработки многопоточных приложений;
- получение синхронизации операций над разделяемыми объектами в условиях многопоточности;
- секции и объекты синхронизации

В ходе работы изучаются следующие классы и интерфейсы платформы J2SE:

- классы и интерфейсы пакета java.lang.* -Runnable, Thread;
- класс объяляемой исключительной ситуации java.lang.Exception;
- классы пакета java.io.* -FileReader, FileWriter, IOException, BufferedReader, FileInputStream, FileOutputStream;

2. Упражнения на программирование

1. Реализовать многопоточное приложение, производящее тестирование целых чисел типа long на простоту в заданном диапазоне. Главный поток приложения формирует пропорциональные последовательности чисел и создает для каждого отдельный поток обработки.

2. Реализовать многопоточное приложение, производящее тестирование целых чисел типа long на простоту в заданном диапазоне. Для этого создается фиксированное количество потоков равное 10-15% от количества элементов, каждый из которых случайным образом обращается к непроверенному элементу последовательности, проверенные элементы удаляются из списка.

3. Провести сравнительную оценку методов реализованных в упражнении 1 и упражнении 2.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
class Base {
    int i;
    Base() { add(1); }
    void add(int v) { i += v; }
```

```
    void print() { System.out.println(i); }
}
class Extension extends Base {
    Extension() { add(2); }
    void add(int v) { i += v*2; }
}
public class Qd073 {
    public static void main(String[] args) {
    bogo(new Extension());
}
static void bogo(Base b) {
    b.add(8);
    b.print();
    }
}
```

2. Что произойдет в результате компиляции и выполнения следующего исходного кода:

```
public abstract class Main {
class TestOutput{
    LinkedList numbers=new LinkedList();
    TestOutput(long count )
        {
            for (long i=0;i<count;i++){
                numbers.add(Math.random());
            }
        }
    public void addNumbers(long count){
        for (long i=0;i<count;i++)
            numbers.add(Math.random());
        }
    public void writeToFile(String fileToWrite)
                            throws IOException
    {
    java.io.FileWriter fw=
            new java.io.FileWriter(fileToWrite);
    try{
```

```
            Iterator i=numbers.iterator();
            while (i.hasNext()){
            Double d=(Double)i.next();
            fw.write(d.toString());
            i.remove();

        }catch (IOException e){
            throw e;
            }
        finally{
            fw.close();
        }

    }
    }
    public static void main(String[] args) {

    try{
    (
    new
TestOutput(Long.parseLong(args[1]))).writeToFile(
                                            args[0]);
    }
    catch (IOException e)
    {
    System.err.print(e);
    }
    }
}
}
```

4.Контрольные вопросы

1. Каким образом ведут себя главный поток приложения и порожденные им потоки, после выполнения всех операций?

2. В чем заключаются отличия между классами TreeSet и HashSet?

3. На основе какой структуры данных реализован класс Hashmap?

Практическое занятие №20

Многопоточные приложения, файловый ввод-вывод, синхронизация

1. Цели работы

Целями практического занятия являются:

- изучение принципов реализации многопоточных приложений;
- получение навыков разработки многопоточных приложений;
- получение синхронизации операций над разделяемыми объектами в условиях многопоточности;

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы и интерфейсы пакета java.lang.* -Runnable, Thread;
- класс объяляемой исключительной ситуации java.lang.Exception;
- классы пакета java.io.* -FileReader, FileWriter, IOException, BufferedReader, FileInputStream, FileOutputStream;

2. Упражнения на программирование

1. Реализовать многопоточное приложение, реализущее поиск подстроки в файлах. Список файлов передается в качестве параметра командной строки. Для каждого файла выделяется отдельный поток. Для вывода результатов поиска в консоль создается отдельный поток, считывающий данные по мере поступления из разделяемого списка объектов класса SearchResult, имеющего следующего поля «имя файла», «индекс вхождения».

2. Реализовать многопоточное приложение, реализующее вывод всех четных слов из списка файлов. Для каждого файла создается новый поток, но общее число потоков не должно превышать 10.

3. Реализовать сравнительную оценку эффективности работы приложений из упражнений 1 и 2, на разном количестве и размере файлов, а также количестве найденных элементов.

3. Задачи

1. Какие утверждения верны для нижеприведенного исходного кода?

```
class Counter {
    int v = 0;
    synchronized void inc() { v++; }
    synchronized void dec() { v--; }
}
public class Q7ed5 {
    Counter i;
    Counter j;
    Counter k;
    public synchronized void a() {
         i.inc();
         System.out.println("a");
         i.dec();
    }
    public synchronized void b() {
         i.inc(); j.inc(); k.inc();
         System.out.println("b");
         i.dec(); j.dec(); k.dec();
    }
    public void c() {
         k.inc();
         System.out.println("c");
         k.dec();
    }
}
```

a) i.v всегда равен 0 или 1;

б) j.v всегда равен 0 или 1;

в) k.v всегда равен 0 или 1;

г) j.v всегда больше или равен k.v в любой момент времени;

д) k.v всегда больше или равен k.v в любой момент времени;

4. Контрольные вопросы

1. Какие средства в Java существуют для представления чисел в различных системах счисления?

2. Какие классы платформы J2SE существуют для поддержки многопоточности?

3. Каким образом обеспечивается синхронизация при работе с объектами-коллекциями?

Практическое занятие №21

Распределенные приложения. Пакет java.net.*.

1. Цели работы

Целью работы является получение навыков разработки распределенных клиент-серверных приложений с поддержкой передачи данных по протоколу TCP/IP.

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы пакета java.net для поддержки передачи данных по протоколу TCP/IP -Socket,ServerSocket,Connecttion,
- классы пакета java.io – FileReader, FileWriter, BufferedReader, IOException,DataInputStream, DataOutputStream;

2. Упражнения на программирование

1. Реализовать приложение эхо-клиент и приложение эхо-сервер на основе протокола TCP/IP.

2. Дополнить приложение из упражнения 1 возвратом приложением сервером справочной информации о подключившемся клиенте.

3. Реализовать многопоточный вариант приложения из упражнения 2, с возвратом справочной информации о текущих подключенных клиентах.

4. Реализовать приложение на основе протокола TCP/IP для организации обмена сообщения между клиентами, подключенными к серверу.

5. Дополнить приложение, реализованное в упражнении 4 функции передачи содержимого файлов (без использования протокола FTP) между подключенными клиентами.

6. Реализовать многопользовательское клиент-серверное приложение эмулирующее работу билинговой системы и обладающее следующими функциональными возможностями:

1. открытие счета;

2. внесение средств на счет;

3. снятие средств со счета;

4. контроль баланса счета.

7. Дополнить упражнение 1 реализовав механизмы аутентификации и авторизации. В качестве аутентифицирующей информации необходимо использовать номер счета и пароль, с длиной не менее 6 символов, хранимый как хэш-значение функции MD5. Данные аутентификации хранятся в файле.

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода?

```
public class Sample {
    public static void main(String[] args) {
        LinkedList lla = new LinkedList();
        LinkedList llb = new LinkedList();
        assert lla.size() == llb.size() : "пустой";
        la.add("Hello");
        assert lla.size() == 1 : "размер";
        lib.add("Hello");
        assert llb.contains("Hello") : "содежит";
        assert lla.get(0).equals(llb.get(0)) :
                                    "'элемент";
        assert lla.equals(llb) : "коллекции";
    }
```

```
}
```

4. Контрольные вопросы

1. Какой общий вид прототипа объявления метода класса?

2. Какой механизм используется для передачи параметров в методы класса?

3. Каким образом можно получить из объекта класса java.lang.LinkedList массив хранимых в нем объектов?

Практическое занятие №22

Распределенные приложения. Пакет java.net.*.

1. Цели работы

Целью работы является изучение принципов передачи данных в сети с использованием протокола UDP/IP, а также классов и интерфейсов платформы J2SE, предназначенных для поддержки этого протокола.

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы и интерфейсы пакета java.net - DatagramPacket, DatagramSocket, DatagramSocketImpl, ConnectException;
- класс объяляемой исключительной ситуации java.lang.Exception;
- java.security.MessageDigest;
- классы пакета java.io -FileReader,FileWriter,BufferedReader;

2. Упражнения на программирование

1. Реализовать приложение эхо-клиент и приложение эхо-сервер на основе протокола UDP/IP.

2. Дополнить приложение из упражнения 1, возвратом приложением сервером справочной информации о подключившемся клиенте.

3. Реализовать многопоточный вариант приложения из упражнения 2, с возвратом справочной информации о текущих подключенных клиентах.

4. Реализовать приложение на основе протокола UDP/IP для организации обмена сообщения между клиентами, подключенными к серверу.

5. Дополнить приложение, реализованное в упражнении 4 функции передачи содержимого файлов (без использования протокола FTP) между подключенными клиентами.

6. Реализовать многопользовательское клиент-серверное приложение эмулирующее работу новостной системы. Приложение-клиент может вносить новость и отвечать на сообщения опубликованные другими пользователями. Сообщения доступны для просмотра всеми пользователями и образуют древовидную структуру.

7. Дополнить упражнение 1 реализовав механизмы аутентификации и авторизации. Реализовать специальный класс Moderator, для модерации ранее опубликованных сообщений.

3. Задачи

1. Какое утверждение верно относительно следующего исходного кода программы?

```
public class SampleClass extends Thread{
    static Object lock1=new Object();
    static Object lock2=new Object();
    static volatile int i1,i2,j1,j2,k1,k2;
    public void run(){while (true)
                      {doIt();check();}}
    void doIt(){
         synchronized(lock1){i1++;}
         j1++;
         synchronized(lock2){k1++;k2++;}
         j2++;
         synchronized(lock1){i2++;}
    }
void check(){
    if (i1!=i2) System.out.println("i");
    if (j1!=j2) System.out.println("j");
    if (k1!=k2) System.out.println("k");
}
```

```
    public static void main(String[] args){
        new SampleClass().start();
        new SampleClass().start();
    }
}
```

Выберите правильный ответ:

а) программа не скомпилируется;

б) нельзя сказать определенно, будут ли напечатаны во время выполнения буквы i,j,k;

в) определенно можно сказать, что ни одна из букв i,j,k не будет напечатана во время выполнения;

г) определенно можно сказать, что буквы i и k никогда не будут напечатаны во время выполнения;

д) определенно можно сказать, что буква k никогда не будет напечатана во время выполнения;

4. Контрольные вопросы

1. Для чего предназначены и как реализуются статические секции инициализации?

2. Какие модификаторы доступа доступны при объявлении метода класса?

3. Как организовать перенаправление стандартного потока вывода в файл?

Практическое занятие №23
Распределенные приложения. Пакет java.net.*.

1. Цели работы

Целью работы является получение навыков разработки распределенных клиент-серверных приложений с поддержкой передачи данных по протоколу TCP/IP

В ходе практического занятия изучаются следующие классы и интерфейсы платформы J2SE:

- классы пакета java.net для поддержки передачи данных по протоколу TCP/IP -Socket,ServerSocket,Connecttion,
- классы пакета java.io – FileReader, FileWriter, BufferedReader, IOException,DataInputStream, DataOutputStream
- класс java.lang.Math;

2. Упражнения на программирование

1. Реализовать распределенное клиент-серверное приложение на основе протокола TCP/IP, обладающее следующими функциональными возможностями:

- приложение устанавливает соединение с сервером и начинает вносить в поток передачи данных последовательность случайных целочисленных значений;
- сервер считывает данные из входного потока и заносит их в некоторый буфер, проверяя на наличие вновь вносимых значений в буфер, если происходит попытка дублирования то дублирующиеся значения отбрасываются. При достижении некоторого порогового объема буфера, данные из него дозаписываются в файл, а содержимое буфера копируется в некоторый другой буфер, а вновь принимаемые значения отыскиваются и в этом буфере;

2. На основе приложения из упражнения 1 реализовать приложение, которое бы помимо передачи данных от клиента к серверу, обладало бы возможностью уведомления клиентов о вновь занесенных значениях другими клиентами, и проверка на дублирование значений производилась бы на стороне клиента.

3. Проанализировать эффективность работы приложений по отношению время обработки/количество обработанных значений/количество допущенных ошибок (появления дубликатов)

3. Задачи

1. Что произойдет в результате компиляции и выполнения следующего исходного кода программы?

```
public class Nesting {
    public static void main(String[] args){
    B.C obj=new B().new C();
    }
}
class A{
        int val;
        A(int v){val=v;}
}
class B extends A{
        int val=1;
        B(){super(2);}
class C extends A{
        int val=3;
        C(){
        super(4);
        System.out.println(B.this.val);
        System.out.println(C.this.val);
        System.out.println(super.val);
        }
    }
}
```

2. Какое утверждение относительного приведенного ниже исходного кода верно?

```
public class Joining{
 static Thread createThread(final int i,
                            final Thread t1){
        Thread t2=new Thread(){
        public void run(){
        System.out.println(i+1);
        try{
            t1.join();
        }catch (InterruptedException e){
        }
        System.out.println(i+2);
}
};
```

```
            System.out.println(i+3);
            t2.start();
            System.out.println(i+4);
            return t2;
    }
    public static void main(String[] args){
    createThread(10,createThread(20,
            Thread.currentThread()));
    }
    }
```

Выберите правильные ответы:

а) Первое число, которое будет напечатано, это 13;

б) Число 14 будет напечатано прежде, чем число 22;

в) Число 24 будет напечатано прежде, чем число 21;

г) Последнее число, которое будет напечатано, это 12;

д) Число 11 будет напечатано прежде, чем число 23;

4. Контрольные вопросы

1. Для каких целей предназначены классы java.io.DataInput-Stream, java.io.DataOutputStream?

2. Каким способом можно считать из стандартного потока чтения строку данных?

3. Какие состояния потока вы знаете?

Практическое занятие №24
Основы разработки распределенных приложений на базе RMI

1. Цели работы

Целями практического занятия являются:

- знакомство с технологией RMI,
- определение, реализация удаленного интерфейса,
- вызов методов удаленного объекта;
- методы обеспечения безопасности в распределенных приложений,

- использование файловых потоков ввода/вывода.

2. Изучаемые классы и интерфейсы платформы J2SE

- классы и интерфейсы пакета java.rmi.* - java.io.Remote, java.io.RemoteException, java.rmi.server.UnicastRemoteObject, ;
- класс пакета java.io.* - java.io.FileReader,java.io.FileWriter, java.io.FileInputStream, java.io.BufferedReader, java.io.IOException;
- класс java.lang.Math;
- java.security.MessageDigest;

3. Упражнения на программирование

1. Реализовать многопользовательское клиент-серверное приложение на основе технологии RMI, эмулирующее работу билинговой системы и обладающее следующими функциональными возможностями. Удаленный объект должен обладать следующими методами

1. открытие счета;
2. внесение средств на счет;
3. снятие средств со счета;
4. контроль баланса счета.

2. Дополнить упражнение 1 реализовав механизмы аутентификации и авторизации. В качестве аутентифицирующей информации необходимо использовать номер счета и пароль, с длиной не менее 6 символов, хранимый как хэш-значение функции SHA. Данные аутентификации хранятся в файле.

3. Задания

1. Что произойдет при попытке компиляции и выполнения следующего исходного кода

```
abstract class AbstractClass{
    private int value;
    public abstract void doAction();
    public int getValue(){
        return this.value;
    }
    public void setValue(final int value){
```

```
        if (value>=0)
             this.value=value;
        }
        }
 class AbstractClassRealization extends
  AbstractClass{
        public void doAction()
        {
        value=new Integer((new
        Double(Math.random())).intValue());
        }
}
```

2. Какая из отмеченных строк в следующем коде может быть раскомментированна, так что код будет по прежнему компилироваться без ошибок?

```
class SampleClass{
        // int width =14; /* Строка А*/
        {
             // area=width*height; /*Строка Б*/
        }
        int width=37;
        {
        //height=11 /* Строка В*/
        }
        int height,area;
        //area =width*height;/* Строка Г */
        {
        //int width=15; /* Строка Д*/
        area=100;
        }
};
```

4. Контрольные вопросы

1. В чем заключается сущность технологии RMI?
2. В каких случаях необходимо применять синхронизацию?
3. Каким образом происходит сравнение объектов реализующих интерфейс Comparable?

Приложение 1. Примерный список вопросов к зачету по курсу “Основы Java-технологий”

1. Язык JAVA как объектно-ориентированный язык программирования.
2. Сущность и основные принципы языка JAVA.
3. Понятие платформы программирования.
4. Платформа J2SE. Основные достоинства, недостатки и особенности языка JAVA.
5. Основные области применения языка Java.
6. Основные понятия языка Java. Классы, объекты, интерфейсы, пакеты.
7. Структура файла исходного кода.
8. Виды приложений на языке Java.
9. Структура приложения в платформе J2SE.
10. Пример приложения на языке Java. Компиляция и отладка приложений.
11. Основные элементы языка. Лексемы. Идентификаторы. Ключевые слова.
12. Основные элементы языка. Константы.
13. Основные элементы языка. Примитивные типы данных –целые, символьный, вещественные, логические.
14. Основные элементы языка. Объявление и инициализация переменных. Время жизни переменных.
15. Операторы языка Java. Приоритет и правила ассоциативности. Порядок вычисления операндов.
16. Оператор присваивания =. Присваивание примитивных значений, присваивание ссылок, многократные присваивания.
17. Арифметические операторы. Приоритет и ассоциативность арифметических операторов.
18. Арифметические операторы. Порядок вычислений в арифметических выражениях.
19. Арифметические операторы. Диапазон числовых значений.
20. Арифметические операторы. Унарные и бинарные арифметические операторы.
21. Арифметические операторы. Составные операторы присваивания. Операторы декремента и инкремента.

22. Операторы отношения. Равенство. Равенство значений примитивных типов данных.
23. Равенство ссылок на объекты. Равенство значений объектов.
24. Булевы логические операторы. Логические составные операторы присваивания. Условные операторы && и || . Условный оператор ?/
25. Целочисленные поразрядные операторы. Операторы ~,&, |,^. Операторы сдвига <<, >>,>>>.
26. Унарный оператор приведения (тип).
27. Преобразования сужения и расширения.
28. Числовые расширения.
29. Примеры преобразования типов. Неявное преобразование типов. Правила приведения.
30. Массивы. Объявление переменных -массивов.
31. Создание массива. Инициализация массива.
32. Использование массивов. Анонимные массивы. Многомерные массивы.
33. Операторы ветвления – сокращенный оператор if.
34. Оператор if –else.
35. Оператор switch.
36. Операторы цикла. Оператор while. Оператор do-while.
37. Оператор цикла for.
38. Команды перехода. Маркированные операторы. Операторы break, continue, return.
39. Определение классов. Модификаторы объявления классов.
40. Структура класса – поля, методы, конструкторы, блоки инициализации.
41. Определение полей. Модификаторы объявления полей. Инициализация значений.
42. Блоки инициализации.
43. Неизменяемые поля.
44. Методы класса. Модификаторы объявления методов классов.
45. Методы класса. Передача параметров.
46. Статические методы.
47. Неизменяемые методы.
48. Абстрактные методы.
49. Модификаторы и правила видимости.
50. Объектная ссылка this.

51. Перегрузка методов.
52. Конструкторы.
53. Конструктор по умолчанию. Перегруженные конструкторы.
54. Блоки инициализации.
55. Объекты. Создание объектов. Время жизни объекта и сборка мусора.
56. Одиночное наследование. Переопределение и скрытие методов.
57. Сокрытие (инкапсуляция) полей;
58. Сокрытие статического метода.
59. Объектная ссылка super. Организация цепочки конструкторов.
60. Интерфейсы.
61. Вложенные классы и интерфейсы.
62. Назначение исключительных ситуации.
63. Типы исключительных ситуаций.
64. Исключительная ситуация как объект.
65. Основные классы исключительных ситуаций. Классы Exception, RuntimeException, Error.
66. Определение новых классов исключительных ситуаций.
67. Обработка исключений –try, catch, finally. Оператор throw. Генерация исключительной ситуации.
68. Диагностические утверждения.
69. Основные классы пакета java.lang. Класс Object.
70. Классы оболочки над примитивными типами.
71. Неизменяемые строки - класс String.
72. Изменяемые строки – класс StringBuffer.
73. Класс Math – реализация основных математических функций.
74. Инструментальный набор коллекций. Классы и интерфейсы для реализации коллекций и карт.
75. Коллекции –основные действия, массовые операции, операции с массивами, итераторы.
76. Множества – классы HashSet и LinkedHashSet.
77. Списки- ArrayList, LinkedList
78. Карты. Классы HashMap, LinkedHashMap и Hashtable.
79. Отсортированные множества и отсортированные карты.
80. Интерфейсы Comparator, Comparable, SortedSet, SortedMap.
81. Работа с коллекциями.
82. Организация чтения/записи байтовых потоков.
83. Классы InputSteamReader и OutputStreamReader.

84. Буферизованные потоки чтения и записи – класс BufferedReader.
85. Чтение и запись из файлов.
86. Организация взаимодействия с консолью.
87. Многозадачность. Обзор потоков.
88. Главный поток.
89. Создание потока, реализация интерфейса Runnable.

Список рекоммендуемой литературы

1. Арнолд К., Гослинг Дж., Холмс Д. Язык программирования Java. М. «Вильямс», 2001 г.
2. Гергель В.П., Свистунов А.Н. "Методы и средства построения распределенных программных систем с использованием технологии Java", Нижегородский Государственный Университет им. Н.И.Лобачевского;
   http://ru.sun.com/research/Gergel_all_materials.zip
3. Вязовик Н.А. Программирование на Java. Курс лекций (гриф УМО)
4. The Java Language Specification.
   http://java.sun.com/docs/books/jls/index.html
5. Халид А. Мугал, Рольф В. Расмуссен. Java. Руководство по подготовке к сдаче сертификационного экзамена СХ-310-035. М. Кудиц-образ, 2006.
6. Хорстманн К. Java 2. Т1. Основы. 7 издание. М: Диалектика, 896 стр.
7. Хорстманн К. Java 2. Т1. Тонкости программирования. 7 издание. М: Вильямс, 1168 стр.
8. Шилдт Г., Холмс Дж. Искусство программирования на Java.