МИНИСТЕРСТВО ОБРАЗОВАНИЯ и НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ

КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И. РАЗЗАКОВА

Кафедра «Информационной системы и технологии»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ по дисциплине
“Теория информационных процессов и систем ” ТИПиС”
форма обучения дневная

Для студентов по направления
710200 Информационной системы и технологии

Бишкек 2014

«Рассмотрено» на заседании кафедры ИСТ прот. № 3 от 7.10.2014г.	«Одобрено» Учебно-методической комиссией ИУиБ прот. №2 от 27.10.2014

Составитель: Боскебеев К.Дж.

Методические указания к выполнению лабораторных работ по дисциплине ТИПиС КГТУ им. И. Раззакова, института управления и бизнеса, института электроники и телекоммуникации, института горного дела и горных технологий им. академика У.Асаналиева. /КГТУ им. И. Раззакова; Сост.: Боскебеев К.Дж./ - Б.: ИЦ «Текник», 2014. - 36 с.

Предлагаемая работа является развёрнутым руководством, содержащим рекомендации по выполнению лабораторных работ. Для студентов по направления **710200 Информационной системы и технологии** Приведены требования к подготовке лабораторных работ, выполнения, порядку защиты их и примеры. Методические указание адресованы не только студентам указанной специальности, но смежным специальностей, например ИВТ и программная инженерия.

Библ. 4 назв.

Рецензент: д.т.н., профессор Батырканов Ж.И.

Оглавление

Лабораторная работа №1

Тема работы. Отладка программы на языке JAVA в операционной системе Windows XP/7 командной строке.

Цель работы. Установка Java Virtual Machine (JVM). Познакомиться со структурой программы, описываемой на языке JAVA. Изучить особенности языка JAVA. Научиться составлять программу. Изучение проводится на нескольких примерах.

Краткие теоретические сведения Java

Java является объектно-ориентированным языком программирования, разработанным фирмой **Sun Microsystems** (сокращенно, Sun).

Основные достоинства языка

- Наибольшая среди всех языков программирования степень переносимости программ.
- Мощные стандартные библиотеки.
- Встроенная поддержка работы в сетях (как локальных, так и Internel/Intranet).

Основные недостатки

- Низкое, в сравнении с другими языками, быстродействие, повышенные требования к объему оперативной памяти (ОП).
- Большой объем стандартных библиотек и технологий создает сложности в изучении языка.
- Постоянное развитие языка вызывает наличие как устаревших, так и новых средств, имеющих одно и то же функциональное назначение.

Основные особенности

- Java является полностью объектно-ориентированным языком. Например, C++ тоже является объектно-ориентированным, но в нем есть возможность писать
  программы не в объектно-ориентированном стиле, а в Java так нельзя.
- Реализован с использованием интерпретации P-кода (байт-кода). Т.е. программа сначала транслируется в машино независимый P-код, а потом интерпретируется некоторой программой-интерпретатором (виртуальная Java-машина, JVM).
- Рассмотрим демонстрационный пример — приложение типа " Hello, World! ". Но сначала выполним некоторые подготовительные действия, которые облегчат нам жизнь в дальнейшем.

Подготовительные действия по установке Java Virtual Machine (JVM) на Windows

Скачайте версию JDK(на время написание этой статьи версия jdk-7u25-windows-i586.exe). Установите JDK, папка, в которую установлено JDK будет

известна как JAVA. Установите JDK по умолчанию c:\Program Files\Java\jdk1.7.0_25. Для проверки установки компилятора Java проделайте следующие действия:

- Вызвать командную строку.
- Набрать команду java и нажать Enter (если вышло описание команды java то установка JDK прошло успешно).
- Набрать команду javac (JAVA Compiler) и нажать Enter (если вышло сообщение о том, что команда не распознана то проделать следующие шаги).

Для этого необходимо на ярлыке «Мой компьютер» нажать правой кнопкой мыши, выбрать «Свойства», откроется окно «Свойства системы», выбрать вкладку «Дополнительно» затем нажать кнопку «Переменные среды». В разделе «Переменные среды пользователя» создать переменную PATH и в поле значение переменной указать путь по которому лежит файл javac (по умолчанию это C:\Program Files\Java\jdk1.7.0_25\bin). Нажимаем ОК. Проверяем появилась ли переменная PATH. Нажимаем ОК. Перегружаем компьютер. Перегрузившись, вызываем командную строку и набираем команду javac, если вышло описание команды javac, то установка компилятора Java произведена успешно.

После инсталляции пакета SDK можно транслировать программу на Java, вызывая компилятор javac. exe из командной строки. Аналогично, для выполнения можно вызывать JVM (виртуальную машину Java) прямо из командной строки — java. exe.

Первая программа

Для подготовки Java-программы подойдет любой текстовый редактор. Существенным является только наличие в нем поддержки длинных имен файлов. В простейшем случае Java-приложение состоит из одного Java-файла. Наберем простейшую Java-программу (файл Hello.java):

```
public class Hello {
public static void main(String[] args) {
System. out. println ("Hello, World! ") ;
                    }                    }
```

Для компиляции программы Hello.java вызовем командную строку Пуск→Все программы → Стандартные → Командная срока. Наберем команду приведенный ниже:

C:\Documents and Settings\admin\javac Hello.java

Если Вы набрали текст правильно, то в результате компиляции на экран ничего не будет выведено. Если же нет, то на экран будут выданы сообщения об ошибках. Транслятор создаст файл **Hello.class** с независимым от процессора байт-кодом нашего примера. Для того чтобы исполнить полученный код, необходимо иметь среду времени выполнения языка Java (в нашем случае это программа Java), в которую надо загрузить новый класс для исполнения. Для выполнения программы **Hello.class** введём команду описанный ниже:

C:\Documents and Settings\admin\java Hello.class

После исполнения программы вы должны получить следующий результат:

Hello World

Полезного сделано мало, однако мы убедились, что установленный Java-транслятор и среда времени выполнения работают.

При этом необходимо набрать имя запускаемого класса точно так, как оно написано в исходном тексте программы, т.е. с соблюдением регистра, иначе вы получите сообщение об ошибке.

Очень распространённой формой представления алгоритмов (в силу ее удобства и понятности) является блок-схема — графическое изображение алгоритма в виде определённым образом связанных между собой нескольких типов блоков.

Пример программ по JAVA приведено ниже.

```
*Оператор деления по модулю
*/
public class OpEquals {public static void main(String args[])  {
  int a = 1; int b = 2; int c = 3; a += 5;    b *= 4; c += a * b;
c %= 6;
System.out.println( "a = " + a);
System.out.println("b = " + b);
System.out.println("c = " + c);
} }
```

А вот и результат, полученный при запуске этой программы:

a = 6 b = 8 c = 3

Следующий пример иллюстрирует использование операторов инкремента и декремента.

```
public class IncDec {
public static void main(String args[])  {
int a = 1;
int b = 2;
int c = ++b;
int d = a++;
c++;
System.out.println("a = " + a);
System.out. println("b = " + b);
System.out.println("c = " + c);
System.out.println("d = " + d);
} }
```

Результат выполнения данной программы будет таким:

a = 2

b = 3

c = 4 d = 1

Теперь, когда мы познакомились со всеми простыми типами, включая целые и вещественные числа, символы и логические переменные, давайте попробуем собрать всю информацию вместе. В приведённом ниже примере создаются переменные каждого из простых типов и выводятся значения этих переменных.

```
public class SimpleTypes {
public static void main(String args [])  {byte b = 0x55; short s = 0x55ff;
int i = 1000000; long l = 0xffffffffL; char c =  'a'   ; float f =  .25f;
double d =  .00001234;       boolean bool = true;
 System.out.println("byte b = " + b);  System.out.println("short s = " +s);
System.out.println("int i = " + i);           System.out.println("long 1 = " + 1);
System.out.println("char c = " + c);  System.out.println("float f = " + f);
System.out.println("double d = " + d);
System.out.println("boolean bool = " + bool);
}}
```

Запустив эту программу, вы должны получить результат, показанный ниже:

```
byte b = 85 short s =22015 int i = 1000000
long 1 = 1 char c = a float f = 0.25 double d = 1.2346-005
 boolean bool = true
```

Обратите внимание на то, что целые числа печатаются в десятичном представлении, хотя мы задавали значения некоторых из них в шестнадцатеричном формате.

Варианты заданий к лабораторной работе №1

1. Вывести на консоль следующий набор символов

```
*
***
*****
*******
```

2. Разделить вещественное число X на Y и результат вывести на экран дисплея.
3. Вычислить остаток от деления действительного числа X на Y и остаток вывести на экран дисплея.
4. Даны два действительных числа X и Y. Возвести в куб X и Y и результат вывести на экран дисплея.

Задание к работе

- Установить программу на языке JAVA.
- Продемонстрировать работу программы на ПК.

Домашняя подготовка. Для допуска к лабораторной работе:

- Изучить (по литературе) тему - структура программ, понятия тип данных. Подготовить ответы на контрольные вопросы.

Форма отчета и порядок защиты. Студент должен:

- Представить отчет о проделанной работе. В отчет должны входить ответы на контрольные вопросы;
- Ответить на контрольные вопросы;

Контрольные вопросы

1. Что такое переменная?
2. Что такое простые типы?
3. Что такое типы с плавающей точкой?
4. Что такое символьный тип?

В этой лабораторной работе мы научились составлять простые программы на языке Java. В следующей лабораторной работе мы научимся работать в платформе Eclipse.

Список использованных источников

1. Боскебеев К.Дж. Программирование на языке Java. Учебное пособие для вузов./ КГТУ им. И Раззакова: – Б.: ИЦ «Техник», 2010. - 116с.

2. Хабибулин И.Ш. Самоучитель Java 2. – СПб.: БХВ-Петербург,2007. - 720с.

Лабораторная работа №2

Тема работы. Введение в среду программирования Eclipse.

Цель работы. Начальное знакомство с IDE Eclipse.

Содержание работы. Рассматриваются базовые компоненты интерфейса пользователя Eclipse. Компоновки, представления. Создание проекта, создание и операции с классами. Eclipse — один из лучших инструментов Java, созданных за последние годы. SDK Eclipse представляет собой интегрированную среду разработки (IDE, Integrated Development Environment) с открытым исходным кодом.

Основные инструментальные средства Eclipse Java включают в себя: редактор исходного кода (создание и редактирование исходного текста программ), средства отладки и интеграции с Ant. Кроме этого в Eclipse доступны множество бесплатных и коммерческих дополнений (плагинов), таких, как инструментальные средства создания схем UML, разработка баз данных и др. Собственно сама по себе Eclipse — это только платформа, которая предоставляет возможность разрабатывать дополнения, называемые плагинами, которые естественным образом встраиваются в платформу В Eclipse доступны дополнения для следующих языков: C/C++, Html, Cobol, Perl, Php, Ruby и др. Вы можете также разработать собственное дополнение для расширения возможностей Eclipse.

Варианты заданий к лабораторной работе №2

1. Установите последовательно компоновки на вашем рабочем столе: Java, Java Browsing, Debug. Укажите функции и назначение этих компоновок.

2. Работа с представлениями. В компоновку Java добавьте новые представления: Problems, Members и затем их закройте. В компоновку Java Browsing добавьте новые представления: Debug, Display, Memory и затем их закройте.

3. Создайте новый проект newProject1. Добавьте в него три пустых класса: wnclass1, wnclass2, wnclass3.

4. Измените место расположение на диске классов поместив их в новый предварительно созданный каталог New.

5. Создайте новый проект Eclipse с именем newClock из примера апплета JDKX.X.X./Demo/Clock.

Задание к работе

- Установить программу Eclipse.
- Продемонстрировать работу программы на ПК.

Домашняя подготовка. Для допуска к лабораторной работе:

- Изучить (по литературе) тему –создание проекта и класса. Подготовить ответы на контрольные вопросы.

Форма отчета и порядок защиты. Студент должен:

- Представить отчет о проделанной работе. В отчет должны входить ответы на контрольные вопросы;
- Ответить на контрольные вопросы;

Контрольные вопросы

1. Укажите, для каких целей предназначена IDE Eclipse.
2. Укажите последовательность действий для вызова панели Outline.
3. Перечислите основные представления, доступные в перспективе Java.
4. Существует ли возможность создать проект Eclipse на основе уже готовых исходных текстов, созданных в других средах разработки? Что для этого необходимо сделать?

В этой лабораторной работе мы научились работать с платформой Eclipse и составлять простые программы на языке Java.

В следующей лабораторной работе мы научимся использовать условный оператор в языке Java на платформе Eclipse.

Список использованных источников

1. Боскебеев К.Дж. Программирование на языке Java. Учебное пособие для вузов./ КГТУ им. И Раззакова: – Б.: ИЦ «Техник», 2010. - 116с.

2. Хабибулин И.Ш. Самоучитель Java 2. – СПб.: БХВ-Петербург,2007. - 720с.

Лабораторная работа №3

Тема работы. Условный оператор.

Цель работы. Познакомиться со структурой программы язык JAVA в среде условного оператора.

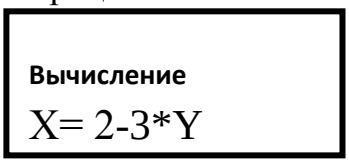
Содержание работы.

Очень распространённой формой представления алгоритмов (в силу ее удобства и понятности) является блок-схема — графическое изображение алгоритма в виде определённым образом связанных между собой нескольких типов блоков.

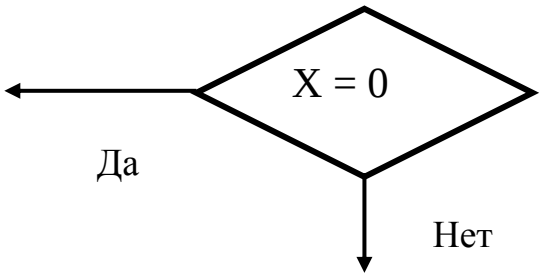
Блоки начала и конца алгоритма изображаются овалами. В блоке начала иногда записывают название блок-схемы.

Блоки ввода и вывода изображаются параллелограммами.

Арифметический блок изображается прямоугольником, внутри которого записывается арифметическая операция.



Логический блок представляется в виде ромба с записанным в нем условием.



Блоки соединяют между собой линиями, обозначающими порядок следования операций. Каждый блок, кроме блоков начала и конца алгоритма, имеет входящие и выходящие из него стрелки. Арифметические блоки снабжаются одним входом и одним выходом, логические—одним входом и двумя выходами, на которых указаны слова «Да» и «Нет», обозначающие выходы при выполнении и невыполнении условия, записанного внутри блока. Требования, которые предъявляются к алгоритмам:

— алгоритм должен быть понятным, что особенно важно в тех случаях, когда приходится читать «чужие» алгоритмы;

— алгоритм должен быть легко проверяемым;

— алгоритм должен допускать возможность его модификации без существенной перестройки всей структуры.

Для достижения указанных свойств при разработке алгоритмов придерживаются особой методики, называемой структурным программированием или структурным подходом. Разработка алгоритмов на основе структурного подхода основывается на использовании трёх типов блоков (вершин): функционального (арифметического), предикатного (логического) и объединяющего (рис. 1, а, б, в),

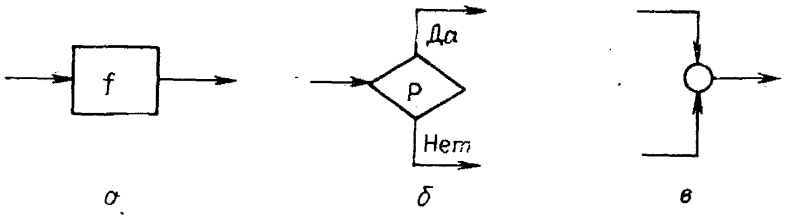


Рис. 1. Блоки (вершины) на блок-схеме:
а — функциональный (вершина); б — предикатный (вершина);
в — объединяющий (вершина)

а также четырёх элементарных блок-схем (рис. 3). Структурная блок-схема— это блок-схема, которая может быть выражена как композиция из четырёх элементарных блок-схем. Любая блок-схема может быть представлена структурной блок-схемой, т. е. для разработки любого алгоритма достаточно четырёх элементарных блок-схем, приведённых на рис. 2.

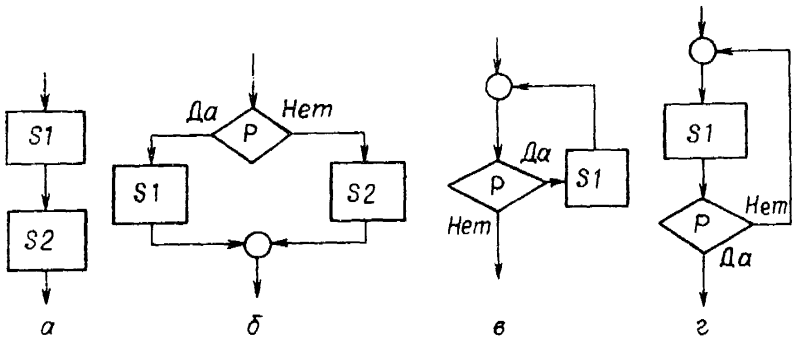


Рис. 2. Четыре элемента структурной блок-схемы

Структура на рис. 2.а и называется следованием, на рис. 2.б — разветвлением, на рис. 2.в—циклом «Пока», на рис. 2.г—циклом «До». Поскольку каждая из этих структур имеет по одному входу и выходу, то и любой алгоритм, «собранный» в любом порядке из этих структур, также будет иметь по одному входу и выходу.

Структура следование означает, что два или более функциональных блока могут быть размещены друг за другом. Структура разветвление применяется, когда в зависимости от условия P нужно выполнить либо одно действие (функциональный блок S1), либо другое действие (функциональный блок S2).

В структуре цикл «Пока» блок S1 размещен после проверки условия P так, что может оказаться, что блок S1 не выполнится ни разу. Здесь P является условием продолжения цикла (всякий раз, когда P истинно, блок S1 выполняется). В структуре цикл «До» (рис. 2, г) блок S1 расположен до проверки логического условия P, так что в этом случае блок S1 выполнится по крайней мере один раз. Здесь P является условием выхода из цикла (как только P становится истинным, выполнение цикла завершается).

Блоки S1 и S2 (рис. 2) могут в свою очередь содержать любые из перечисленных элементарных блок-схем. Это позволяет строить как угодно сложные алгоритмы, развивая их не только «вширь», но и «вглубь».

Конструируемые таким образом алгоритмы имеют четкую и ясную структуру, легко поддаются проверке, так как состоят из ограниченного числа различных блоков, устроенных аналогично. Рассмотрим небольшой пример:

Структурную блок-схему алгоритма поиска большего из трех чисел можно изобразить следующим образом (рис.3). Алгоритм использует промежуточную переменную памяти для временного хранения найденного максимального значения.

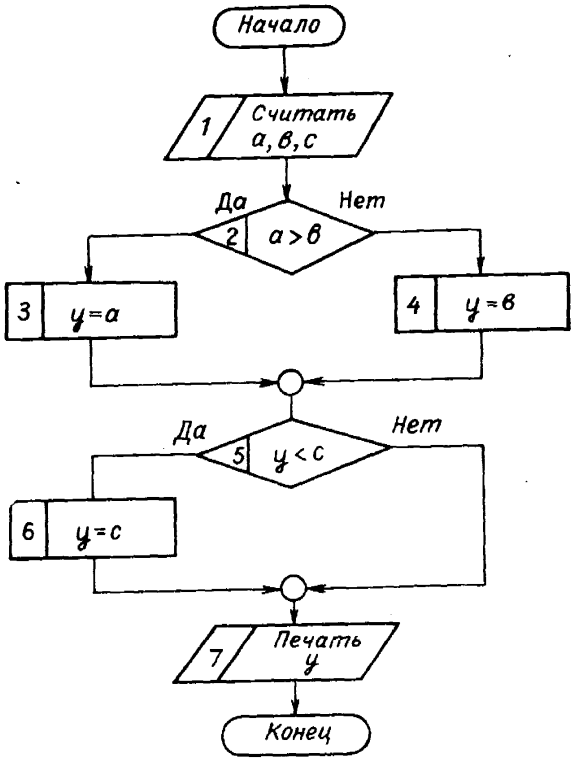


Рис. 3. Структурная блок-схема алгоритма поиска большего из трёх чисел.

Ниже приведена программа на языке JAVA определения максимального значения из трёх чисел.

```
/* Поиск большего из трёх чисел
public class Max {
public static void main(String args[]) {
int a=1;
int b=4;
```

```
int c=3;
if(a>b){ y=a;}
else {y=b; }
if(y<c) { y=c;}
System.out.println("y="+y);
} }
```

После выполнения программы вы должны получить следующий результат:

y=4

Далее приведено варианты заданий к лабораторной работе №2.

Варианты заданий к лабораторной работе №3

1. Даны действительные числа a,в,c . Удвоить эти числа если a≥в≥c, и заменить их абсолютными значениями, если это не так.
2. Даны три действительных числа X,У (X≠У). Меньшее из этих двух чисел заменить их полусуммой, а большее их удвоенным произведением.
3. Даны три действительных числа. Выбрать из них те, которые принадлежат интервалу (1,3).
4. Даны три действительных числа. Выбрать из них те, которые принадлежат интервалу (1,3).
5. Даны три действительных числа. Возвести в квадрат те из них, значения которые неотрицательны.
6. Если сумма трех попарно различных действительных чисел X,Y,Z меньше единицы, то наименьшее из этих трех чисел заменить полу суммой двух других; в противном случае заменить меньшие из X и У полу суммой двух оставшихся значении.
7. Даны действительные числа a,в,c,d если a≤в≤c≤d , то каждое число заменить наибольшим из них; если a>b>c>d , то числа оставить без изменения; противном случае все числа заменяются их квадратами.
8. Даны действительные числа X,У. Если X и У отрицательны, то каждое значение заменить его модулем; если отрицательно только одно из тех, то оба значения увеличить на 0,5 ; если оба значения неотрицательны и ни одно из них не принадлежит отрезку (0,5;2,0), то оба значения уменьшить в 10 раз ; в остальных случаях X и У оставить без изменения.
9. Даны действительные положительные числа X,У,Z. Выяснить, существует ли треугольник с длинами сторон X,У,Z. Если треугольник существует, то ответить - является ли он прямоугольником.
10. Даны действительные числа a,в,c (a ≠0) Выяснить , имеет ли уравнение ax^2+вх+c=0 действительные корни. Если действительные корни имеются, то найти их. В противном случае ответом должно служить сообщение, что действительных корней нет. Даны действительные числа a,в,c (a ≠0) Выяснить , имеет ли уравнение ax^2+вх+c=0 действительные корни. Если действительные корни имеются, то найти их. В противном случае ответом должно служить сообщение, что действительных корней нет.

11. Дано действительное число h. Выяснить, имеет ли уравнение $ax^2 + bx + c = 0$ действительные корни, если:

12. Если действительные корни существуют то найти их. В противном случае ответом должно служить сообщение, что действительных корней нет.

13. Даны действительные числа X_1, X_2, X3,Y_1,Y_2,Y_3.Принадлежит ли начало координат треугольнику с вершинами (X_1,Y_1),(X_2,Y_2),(X_3,Y_3)?

14. Дан треугольник со сторонами a,в,c. Составить программу для определения вида треугольника и печати соответствующего сообщения.

15. Даны произвольные отрезки a,в,c и d. Составить программу определения возможности построения из них параллелограмма и печати соответствующего сообщения.

16. Найти остаток от делителя целой части значения функции $y = \ln(x^2 + ab)$ на 7 и в зависимости от его величины напечатать сообщение об одном из дней недели, пронумеровав их от 0 до 6.

17. Составить программу нахождения наибольшего и наименьшего их трех произвольных чисел.

18. Даны три числа A,B,C. Если все числа положительные вычислить *Z* = A+B+C, если все отрицательные - Z = (A+B)*C, в противном случае Z=A*B*C

19. Составить программу , печатающую одно из сообщений: “ Корни действительные и равные”,"Корни действительные и различные", "Корни мнимые", в зависимости от вида корней уравнения $ax^2 + bx + c = 0$ Принять a,b,c≠0

Задание к работе и порядок выполнения

- Составить программу для заданного варианта;
- Записать программу на языке JAVA;
- Отладить программу;
- Продемонстрировать работу программы на ПК для заданного варианта.

Домашняя подготовка. Для допуска к лабораторной работе:

- Изучить (по литературе) тему - структура программ, формы условного оператора. Подготовить ответы на контрольные вопросы.

Форма отчета и порядок защиты. Студент должен:

- Представить отчет о проделанной работе. В отчет должны входить ответы на контрольные вопросы и текст программы;
- Ответить на контрольные вопросы;
- Выполнить дополнительный вариант задания по указанию преподавателя.

Контрольные вопросы

1. Что такое логический тип?
2. Каковы формы условного оператора? Нарисуйте его блок - схемы.
3. Как записываются на JAVA арифметические функции?
4. Что такое логическое выражение? Из чего оно состоит?

5. Что такое отклонение? Какие операции отношения Вам известны?
6. Какие логические операции применяются в Java? Объясните их действие.
7. Каков порядок вычисления значения логического выражения?
8. Приведите примеры сложных условных выражений.
9. Что такое **break?**
10. Что такое оператор switch?

В этой лабораторной работе мы научились составлять программы на языке Java в платформе Eclipse. В следующей лабораторной работе мы научимся применять оператор цикла FOR.

Список использованных источников

1. Боскебеев К.Дж. Программирование на языке Java. Учебное пособие для вузов./ КГТУ им. И Раззакова: – Б.: ИЦ «Техник», 2010. - 116с.
2. Хабибулин И.Ш. Самоучитель Java 2. – СПб.: БХВ-Петербург,2007. - 720с.

Лабораторная работа №4

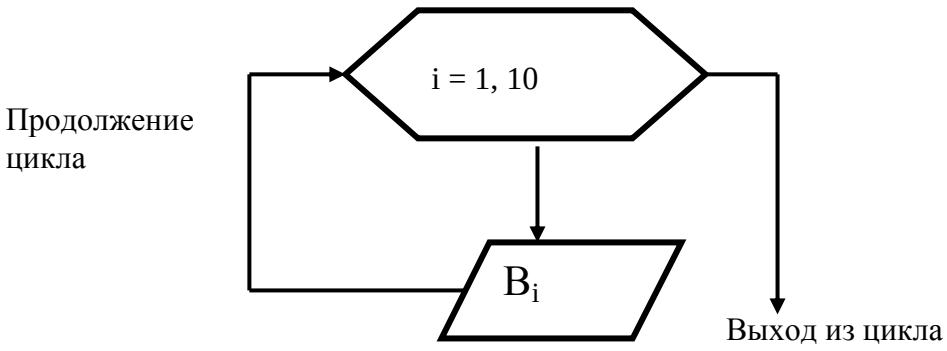
Тема работы. Циклы.

Цель работы. Научиться составлять программу с использованием оператора цикл **FOR** в среде Eclipse.

Материальное обеспечение.

Компьютерное оборудование.

Пример: Блока параметрического цикла. Вводим 10 элементов в массив B[i]



/* В канонической форме цикла **for** происходит увеличение целого значения счётчика /* с минимального значения до определённого предела (в данном примере до 10)

```
public class ForDemo  {
public static void main (String args[])  {
 for (int i=1;   i <= 10;   i++)
System.out.println("i =  "+ i);
} }
```

После выполнения программы вы должны получить следующий результат:

```
i= 1
i= 2
- - - -
i= 10
```

Далее приведено варианты заданий к лабораторной работе №3.

Оператор запятая

Иногда возникают ситуации, когда разделы инициализации или итерации цикла **for** требуют нескольких операторов. Поскольку состав ной оператор в фигурных скобках в заголовок цикла **for** вставлять нельзя, Java предоставляет альтернативный путь. Применение запятой (,) для разделения нескольких операторов допускается только внутри круглых скобок оператора **for.** Ниже приведен тривиальный пример цикла **for,** в котором в разделах инициализации и итерации стоит несколько операторов [2].

```
class Comma  {
public static void main(String args[])  {
int a,   b;
for (a = 1,   b = 4;   a <  b;   a++,   b--)  {
System.out.println("a = " + a);
System.out.println("b = " + b);
} }}
```

Вывод этой программы показывает, что цикл выполняется всего два раза.

```
C:\Documents and Settings\admin>java Comma
a = 1 b = 4 a = 2 b = 3
```

continue

В некоторых ситуациях возникает потребность досрочно перейти к выполнению следующей итерации, проигнорировав часть операторов тела цикла, еще не выполненных в текущей итерации. Для этой цели в Java предусмотрен оператор continue. Ниже приведен пример, в котором оператор continue используется для того, чтобы в каждой строке печатались два числа [2].

```
class ContinueDemo {
public static void main(String args[])  {
for (int i=0;   i <  10;   i++)  {
System.out.print(i + " ");
if (i % 2 == 0) continue;
System. out. println("");
} } }
```

Если индекс четный, цикл продолжается без вывода символа новой строки. Результат выполнения этой программы таков:

```
C:\Documents and Settings\admin>Java ContlnueDemo
0 1
2 3
4 5
6 7
8 9
```

Как и в случае оператора break, в операторе continue можно задавать метку, указывающую, в каком из вложенных циклов вы хотите досрочно прекратить выполнение текущей итерации.

Для иллюстрации служит программа, использующая оператор continue с меткой для вывода треугольной таблицы умножения для чисел от 0 до 9 [2]:

```
class ContinueLabel  {
public static void main(String args[])  {
outer:      for (int  i=0;   i < 10;   i++)  {
                        for (int j  = 0;   j  < 10;   j++) {
                           if (j  > i)  {
                         System.out.println("");
                           continue outer;
                        }
 System.out.print("  "  + (i*j));
}}}}
```

Оператор continue в этой программе приводит к завершению внутреннего цикла со счетчиком] и переходу к очередной итерации внешнего цикла со счетчиком i. В процессе работы эта программа выводит следующие строки:

```
C:\Documents and Settings\admin>Java ContinueLabel
0
01
02 4
03  6  9
04 8   12   16
05  10   15  20  25
06  12   18  24  30 36
07  14 21   28  35 42 49
08   16  24  32 40 48   56  64
09   18  27  36 45  54 63  72  81
```

Варианты заданий к лабораторной работе №4

1. В одномерном массиве, состоящим из *n* вещественных элементов, вычислить: Сумму отрицательных элементов массива;Произведение элементов массива, расположенных между максимальным и минимальным элементами.Упорядочиь элементы массива по возрастанию.

2. В одномерном массиве, состоящим из *n* вещественных элементов, вычислить: Сумму положительных элементов массива; Произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.Упорядочить элементы массива по убыванию.

3. В одномерном массиве, состоящим из *n* вещественных элементов, вычислить: Произведение элементов массива с четными номерами;Сумму элементов массива,расположенных между первым и последними элементами.Преобразовать массив таким образом, чтобы сначала располгались все положительные элементы, а потом – все отрицательные (элементы, равные нулю, считать положительными).

4. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Сумму элементов массива с нечетными номерами;Сумму элементов массива, расположенных между первым и последним отрицательными элеметами.Сжать массив, удалив из него все элементы, модуль которых не превышает единицу. Освободившиеся в конце массива элементы заполнить нулями.

5. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Максимальный элемент массива;Сумму элементов массива, расположенных до последнего положительного элемента.Сжать массив, удалив из него все элементы, модуль которых находится в интервале $[a,b]$. Освободившиеся в конце массива элементы заполнить нулями.

6. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Минимальный элемент массива;Сумму элементов массива, расположенных между первым и последним положительными элементами.Преобразовать массив таким образом, чтобы сначала располагались все элементы. Равные нулю, а потом – все остальные.

7. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Номер максимального элемента массива;Произведение элементов массива, расположенных между первым и вторым нулевыми элементами.Преобразовать массив таким образом, чтобы в первой его половине располагались элементы, стоявшие в неченных позициях, а во второй половине –элементы, стоявшие в четных позициях.

8. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Номер минимального элемента массива;Сумму элементов массива, расположенных между первым и вторым положительными элементами.Преобразовать массив таким образом, чтобы сначала располагались все элементы, модуль которых не рпевышает единицу, а потом – все остальные.

9. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Максимальный по модулю элемент массива; Сумму элементов массива, расположенных между первым и вторым положительными элементами.Преобразать массив таким образом, чтобы элементы равные нулю, располагались после всех остальных.

10. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Минимальный по модулю элемент массива;Сумму элементов массива,расположенных после первого элемента, равного нулю.Преобразать массив таким образом, чтобы в первый его половине располагались элементы, стоявшие в четных позициях, а во второй половине –элементы, стоявшие в нечетных позиыциях.

11. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Номер минимального по модулю элемента массива;Сумму модулей элементов массива, расположенных после первого отрцательного элемента.Сжать массив, удалив из него все элементы, величина которых находится в интервале $[a,b]$. Освободившиеся в конце массива элементы заполнить нулями.

12. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Номер максимального по модулю элемента массива;Сумму элементов массива, расположенных после первого положительного элемента. Преобразать массив таким образом, чтобы сначала располагались все элементы, целая часть которых лежит в интервале $[a,b]$, а потом – все остальные.

13. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Количество элементов массива, лежащих а диапазоне от A до B;Сумму элементов массива, расположенных после максимального элемента. Упорядочить элементы массива по убыванию модулей.

14. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Количество элементов массива, равных нулю;Сумму элементов массива, расположенных после миниамльного элемента. Упорядочить элементы массива по возрастанию модулей.

15. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Количество элементов массива, больших C;Произведение элементов массива, расположенных после максимального по модулю элемента. Преобразовать массив таким образом, чтобы сначала распологались все отрицательные элементы, а потом – все положительные (элементы, равные нулю, считать положительными).

16. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Количество отрицательных элементов массива;Сумму модулей элементов массива, расположенных после минимального по модулю элемента.Заменить все отрицательные элементы массиваих квадратами и упорядочить элементы массива по возрастанию.

17. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Количество положительных элементов массива;Сумму элементов массива, расположенных после последнего элемента, равного нулю. Преобразовать массив таким образом, чтобы сначала располагались све элементы, целая часть которых не превышает единицу, а потом – все остальные.

18. В одномерном массиве, состоящим из n вещественных элементов, вычислить: Количество элементов массива, меньших C;Сумму целых частей элементов массива, расположенных после последнего отрицательного элемента.Преобразовать массив таким образом, чтобы сначала располагались все элементы, отличающиеся от максимального не более чем на 20%, а потом – все остальные.

19. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Произведение отрицательных элементов массива;Сумму положительных элементов массива, расположенных до максимального элемента.Изменить порядок следования элементов в массиве на обратный.

20. В одномерном массиве, состоящим из n вещественных элементов, вычислить:Произведение положительных элементов массива;Сумму элементов массива, расположенных до минимального элемента.Упорядочить по возрастанию отдельно элементы, стоящие на четных местах, и элементы, стоящие на нечетных местах.

Задание к работе и порядок выполнения

- Составить программу для заданного варианта;
- Записать программу на языке JAVA;
- Отладить программу;
- Продемонстрировать работу программы на ПК для заданного
- варианта.

Домашняя подготовка. Для допуска к лабораторной работе:

- Изучить (по литературе) тему - формировать простой ввод и вывод данных. Подготовить ответы на контрольные вопросы.

Форма отчета и порядок защиты. Студент должен:

- Представить отчет о проделанной работе. В отчет должны входить ответы на контрольные вопросы и текст программы;
- Ответить на контрольные вопросы;
- Выполнить дополнительный вариант задания по указанию преподавателя.

Контрольные вопросы:

1. Как реализовать оператор цикла?
2. Как работать с двухмерными массивами?
3. В каком файле разместить этот класс.
4. Какие поля должен содержать этот класс.
5. Какие методы должны быть в нем реализованы.
6. Как загрузить программу пользователя для его выполнения?
7. Что такое многомерные массивы?
8. Как объявляется тип массива?
9. Что такое приоритеты операторов?
10. Какие циклы есть в JAVA?

В этой лабораторной работе мы научились составлять программы с помощью оператора **for** на языке Java.

Список использованных источников

3. Боскебеев К.Дж. Программирование на языке Java. Учебное пособие для вузов./ КГТУ им. И Раззакова: – Б.: ИЦ «Техник», 2010. - 116с.

4. Хабибулин И.Ш. Самоучитель Java 2. – СПб.: БХВ-Петербург,2007. - 720с.

Лабораторная работа №5

Тема работы. Введение в объектно-ориентированное программирование (ООП).

Цель работы. Изучение основ ООП на языке JAVA

Содержание работы. В работе рассматриваются приложения, демонтирующие основные принципы ООП.

Базовым элементом объектно-ориентированного программирования в языке Java является класс [1]. Классы в Java не обязательно должны содержать метод **main.** Единственное назначение этого метода — указать интерпретатору Java, откуда надо начинать выполнение программы. Для того, чтобы создать класс, достаточно иметь исходный файл, в котором будет присутствовать ключевое слово **class, и** вслед за ним — допустимый идентификатор и пара фигурных скобок для его тела.

```
class Point {
}
```

Имя исходного файла Java должно соответствовать имени хранящегося в нем класса. Регистр букв важен и в имени класса, и в имени файла.

Класс определяет структуру объекта и его методы, образующие функциональный интерфейс. В процессе выполнения Java-программы система использует определения классов для создания представителей классов. Представители являются реальными объектами. Термины «представитель», «экземпляр» и «объект» взаимозаменяемы. Ниже приведена общая форма определения класса.

```
class имя_класса extends имя_суперкласса  {
type переменная1_обьекта:
type переменная2_объекта:
type переменнаяN_обьекта:
type имяметода1(список_параметров)  {  тело метода;
)
type имяметода2(список_параметров)  (  тело метода;
)
type имя методаM(список_параметров)  { тело метода;
}
}
```

Ключевое слово **extends** указывает на то, что «имя_класса» — это подкласс класса «имя_суперкласса». Во главе классовой иерархии Java стоит единственный ее встроенный класс — **Object.** Если вы хотите создать подкласс непосредственно этого класса, ключевое слово **extends** и следующее за ним имя суперкласса можно опустить — транслятор включит их в наше определение автоматически. Примером может служить класс Point.

Переменные представителей (instance variables)

Данные **инкапсулируются** в класс путем объявления переменных между открывающей и закрывающей фигурными скобками, выделяющими в определении класса его тело. Эти переменные объявляются точно так же , как объявлялись локальные переменные в предыдущих примерах. Единственное отличие состоит в том, что их надо объявлять вне методов, в том числе вне метода main. Ниже приведен фрагмент кода, в котором объявлен класс Point с двумя переменными типа int.

```
class Point { int x,  y;
}
```

В качестве типа для переменных объектов можно использовать как любой из простых типов, так и классовые типы. Скоро мы добавим к приведенному выше классу метод main, чтобы его можно было запустить из командной строки и создать несколько объектов.

Оператор new

Оператор new создает экземпляр указанного класса и возвращает ссылку на вновь созданный объект. Ниже приведен пример создания и присваивание переменной p экземпляра класса Point.

Point p = new Point();

Вы можете создать несколько ссылок на один и тот же объект Приведенная ниже программа создает два различных объекта класса Point и в каждый из них заносит свои собственные значения. Оператор точка используется для доступа к переменным и методам объекта [2].

```
class TwoPoints {
public static void main(String args[]) {
Point p1 = new Point();
Point p2 = new Point();
p1.x = 10;
p1.y = 20;
p2.x = 42;
p2.y = 99;
System.out.println("x = " + p1.x + " y = " + p1.y);
System.out.println("x = " + p2.x + " y = " + p2.y);
} }
```

В этом примере снова использовался класс Point, было создано два объекта этого класса, и их переменным x и y присвоены различные значения.

Таким образом мы продемонстрировали, что переменные различных объектов независимы на самом деле. Ниже приведен результат, полученный при выполнении этой программы.

```
C:\Documents and Settings\admin>Java TwoPoints
x = 10 y = 20 x = 42 y = 99
```

Поскольку при запуске интерпретатора мы указали в командной строке не класс Point, а класс TwoPoints, метод main класса Point был полностью проигнорирован. Добавим в класс Point метод main и, тем самым, получим законченную программу.

```
class Point  {  int x,   y;
public static void main(String args[])  {
Point  p = new PointO;
p.x = 10;
p.У = 20;
System.out.println("x =    " + p.x + " y = " + p.y);
}  }
```

Объявление методов

Методы — это подпрограммы, присоединенные к конкретным определениям классов. Они описываются внутри определения класса на том же

уровне, что и переменные объектов. При объявлении метода задаются тип возвращаемого им результата и список параметров. Обшая форма объявления метода такова: тип имя_метода (список формальных параметров) {
тело метода:
}

Тип результата, который должен возвращать метод может быть любым, в том числе и типом void — в тех случаях, когда возвращать результат не требуется. Список формальных параметров — это последовательность пар тип-идентификатор, разделенных запятыми. Если у метода параметры отсутствуют, то после имени метода должны стоять пустые круглые скобки.

```
class Point { int x, y;
void init(int a, int b) {
 x = a; y = b;
}}
```

Вызов метода

В Java отсутствует возможность передачи параметров по ссылке на примитивный тип. В Java все параметры примитивных типов передаются по значению, а это означает, что у метода нет доступа к исходной переменной, использованной в качестве параметра. Заметим, что все объекты передаются по ссылке, можно изменять содержимое того объекта, на который ссылается данная переменная.

Скрытие переменных представителей

В языке Java не допускается использование в одной или во вложенных областях видимости двух локальных переменных с одинаковыми именами. Интересно отметить, что при этом не запрещается объявлять формальные параметры методов, чьи имена совпадают с именами переменных представителей. Давайте рассмотрим в качестве примера иную версию метода **init,** в которой формальным параметрам даны имена x и y, а для доступа к одноименным переменным текущего объекта используется ссылка **this.**

```
class Point { int x; int y;
void init(int x, int y) {
this.x = x;
this.y = y;
}}
class TwoPointsInit {
public static void main(String args[]) {
Point p1 = new Point();
Point p2 = new Point();
p1.init(10,20);
p2.init(42,99);
System.out,println('x = " + pl.x + " y = “ + pl.y):
System.out.println('x = " + p2.x + " y = “ + p2.y);
 } }
```

Конструкторы

Инициализировать все переменные класса всякий раз, когда создаётся его очередной представитель — довольно утомительное дело да же в том случае, когда в классе имеются функции, подобные методу **init.** Для этого в Java предусмотрены специальные методы, называемые конструкторами. Конструктор — это метод класса, который инициализирует новый объект после его создания. Имя конструктора всегда совпадаг с именем класса, в котором он расположен (также, как и в C++). У конструкторов нет типа возвращаемого результата — никакого, даже **void**. Заменим метод **init** из предыдущего примера конструктором.

```
class Point  {  int x,   y;
 Point(int x,   int y)  {
this.x = x; this.y = y;
}}
class PointCreate {
public static void main(String args[]) {
Point p = new Point(10,20);
System.out. println("x = " + p.x + " y = " + p.y);
 } }
```

Программисты на Pascal (Delphi) для обозначения конструкта используют ключевое слово constructor.

Совмещение методов

Язык Java позволяет создавать несколько методов с одинаковыми именами, но с разными списками параметров. Такая техника называется совмещением методов (method overloading). В качестве примера приведена версия класса **Point,** в которой совмещение методов использовано для определения альтернативного конструктора, который инициализирует координаты x и y значениями по умолчанию (-1) [2].

```
class Point  {  int x,   y;
Point(int x,   int y)  {
this.x = x; this.y = y;
}
Point() {
x = -1;
y = -1;
} }
class PointCreateAlt {
public static void main(String args[])  {
Point p = new Point();
System.out.println("x =  " + p.x +  " y =  " + p.y); } }
```

В этом примере объект класса **Point** создается не при вызове первого конструктора, как это было раньше, а с помощью второго конструктора без параметров. Вот результат работы этой программы:

C:\Documents and Settings\admin>Java PolntCreateAlt

x = -1 y = -1

Решение о том, какой конструктор нужно вызвать в том или ином случае, принимается в соответствии с количеством и типом параметров, указанных в операторе **new.** Недопустимо объявлять в классе методы с одинаковыми именами и сигнатурами. В сигнатуре метода не учитываются имена формальных параметров учитываются лишь их типы и количество.

this в конструкторах

Очередной вариант класса **Point** показывает, как, используя **this** и совмещение методов, можно строить одни конструкторы на основе других.

```
class Point { int x,   y; Point(int x,  int y)  { this.x = x; this, y = y;
PointO  { this(-1,   -1);
}        }
```

В этом примере второй конструктор для завершения инициализации объекта обращается к первому конструктору.

Методы, использующие совмещение имен, не обязательно должны быть конструкторами. В следующем примере в класс **Point** добавлены два метода **distance.** Функция distance возвращает расстояние между двумя точками. Одному из совмещенных методов в качестве параметров передаются координаты точки x и y. другому же эта информация передается в виде параметра-объекта **Point [2].**

```
class Point  {  int x,   y;
Point(int x, int y) {
this.x = x;
this.y = y;
}
double distance(int x, int y) {
int dx = this.x - x;
int dy = this.y - y;
return Math.sqrt(dx*dx + dy*dy);
}
double distance(Point  p) {
 return  distance(p.x, p.y);
}}
```

```
class PointDist {
public static void main(String args[]) {
Point p1 = new Point(0, 0);
Point p2 = new Point(30, 40);
System.out.println("p1 = " + p1.x + ", " + p1.y);
 System.out.println("p2 = " + p2.x + ", " + p2.y);
 System.out.println("p1.distance(60, 80) = " + p1.distance(60, 80));
}}
```

Обратите внимание на то как во второй форме метода distance; получения результата вызывается его первая форма. Ниже приведен результат работы этой программы:

```
C:\Documents and Settings\admin>Java PointDist
 p1  =   0,   0
 p2  =  30,  40
 p1.distance(60,   80)  =  100.0
```

Пример: Создать класс Tv и запустить через TvTest

```
public class TvTest {
        public static void main(String args[]){
            Tv tv = new Tv();
            tv.channelUp();
            System.out.println();
            tv.channelDown();
        }
    }

public class Tv {

    int channel = 10;

    void channelUp(){
        int volume = 10;
        System.out.println("channel of chanelUp() -" + channel);
        System.out.println("volume  of chanelUp() -" + volume);
    }
    void channelDown(){
        int volume = 20;
        System.out.println("channel of chanelDown() -" + channel);
        System.out.println("volume of  chanelDown() -" + volume);
    }
}
```

В процессе работы эта программа выводит следующие строки:

```
channel of chanelUp() – 10
channel of chanelDown() -  10
volume of  chanelDown() -  20
```

Варианты заданий к лабораторной работе №5

1. Создайте приложение, в котором имеются три класса: *dclass1, dclass2, dclass3*. В классе *dclass1*содержится метод main(). Класс *dclass3* наследуется от *dclass1*, а *dclass* – от *dclass3*.
2. Создайте приложение, в котором имеются два класса: *dclass1, dclass2.* В классе *dclass1*содержится метод main(). Создайте методы в *dclass2* для доступа в закрытом переменным класса *dclass1*.

Задание к работе и порядок выполнения

- Составить программу для заданного варианта;
- Записать программу на языке JAVA.;
- Отладить программу;
- Продемонстрировать работу программы на ПК для заданного
- варианта.

Домашняя подготовка. Для допуска к лабораторной работе:

- Изучить (по литературе) тему - объектно – ориентированное программирование. Подготовить ответы на контрольные вопросы.

Форма отчета и порядок защиты. Студент должен:

- Представить отчёт о проделанной работе. В отчёт должны входить ответы на контрольные вопросы и текст программы;
- Ответить на контрольные вопросы;
- Выполнить дополнительный вариант задания по указанию преподавателя.

Контрольные вопросы:

1. Что такое наследование?
2. Что такое инкапсуляция?
3. Что такое полиформизим?

В этой лабораторной работе мы научились составлять классы на языке Java.

Список использованных источников

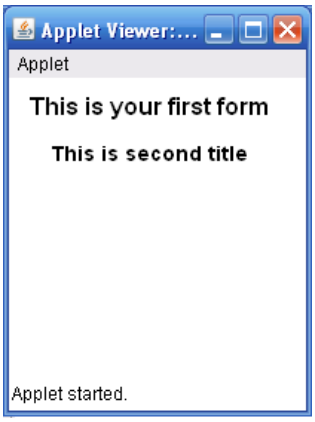
1. Боскебеев К.Дж. Программирование на языке Java. Учебное пособие для вузов./ КГТУ им. И Раззакова: – Б.: ИЦ «Техник», 2010. - 116с.
2. Хабибулин И.Ш. Самоучитель Java 2. – СПб.: БХВ-Петербург,2007. - 720с.

Лабораторная работа № 6

Тема работы. Апплеты.

Цель работы. Знакомство с элементами разработки - апплетов на языке Java.

Программа 1.




```
import java.awt.Color;
import java.awt.Font;
import java.awt.Label;


public class Pig extends Applet{


    public void init()
    {
```

```
        Label header= new Label ("This is your first form");
        header.setFont(new Font("Courier", Font.BOLD,16));

        Label subheader= new Label ("This is second title");
        subheader.setFont(new Font("Courier", Font.BOLD,14));

        header.setForeground(Color.black);
        header.setBackground(Color.white);
        add(header);

        subheader.setForeground(Color.black);
        subheader.setBackground(Color.white);
        add(subheader);
        }
      }
```

Программа 2.

```
import java.applet.*;
import java.awt.*;
public class button extends Applet{

        static String str=new String ("red");
        Button b1;

        public void init()
        {
                b1= new Button ("");
                b1.setLabel(str);
                b1.setBackground(Color.red);
                b1.setForeground(Color.black);
                add(b1);
                }
}
```

Варианты заданий к лабораторной работе №6

1. Создайте форму размером 200x200.
2. Создайте апплет, в котором при нажатии на кнопку выводится матрица 2x2. Используйте метод ***drawstring().***

Задание к работе и порядок выполнения

- Составить программу для заданного варианта;
- Записать программу на языке JAVA.;
- Отладить программу;
- Продемонстрировать работу программы на ПК для заданного варианта.

Домашняя подготовка. Для допуска к лабораторной работе:

- Изучить (по литературе) тему -апплеты. Подготовить ответы на контрольные вопросы.

Форма отчёта и порядок защиты. Студент должен:

- Представить отчёт о проделанной работе. В отчёт должны входить ответы на контрольные вопросы и текст программы;
- Ответить на контрольные вопросы;
- Выполнить дополнительный вариант задания по указанию преподавателя.

Контрольные вопросы:

1. Что такое апплет?
2. Что такое ***drawstring().***?
3. Что такое ***setBackground***?

В этой лабораторной работе мы научились составлять программы апплеты Java. Далее рассмотрим ANDROID+JAVA для сотового телефона

Список использованных источников

1. Ноутон П.,Шилдт Г. Java 2:Пер. с англ. – СПб.:БХВ-Петербург,2007. – 1072с.
2. Хабибулин И.Ш. Самоучитель Java 2. – СПб.: БХВ-Петербург,2007. -720с.
3.

Лабораторная работа № 7

Тема работы. Работа платформе Eclipse ANDROID (ADT).

Цель работы. Разработка программы на языке JAVA в платформе ECLIPSE ANDROID

Краткие теоретические сведения. Запускаем Eclipse (<ADT>\eclipse\eclipse.exe). При первом запуске он попросит указать ему рабочий каталог, где он будет хранить информацию о проектах. Предлагаю создать каталог (<Рабочий стол>\program) и указать этот каталог. Для того, чтобы тестировать приложения, нам понадобится Android Virtual Device (AVD). Это эмулятор Android-смартфона, на который Eclipse сможет устанавливать, созданные нами приложения, и запускать их там. Давайте его создадим. Запускаем Eclipse. Идем в меню Windows > Android Virtual Devices Manager . Жмем кнопку New. В списке Target представлены платформы Android различных версий. Слева выбираете нужную версию и смотрите информацию по ней. Вернемся в Eclipse. Давайте выберем из списка платформу 4.2.2 (API Level 17). Соответственно имя зададим – AP17.Device =3.2”QWGA(ADP2) (320x480 mdpi)

Target = Android 4.2.2 API Level 17. Memory Options -> RAM= 512 -> VM Heap = 64

OK -> Click g по строке активизируются кнопки выберем кнопку Start -> Launch.

Запускается, эмулятор сотового телефона, подождём минут 4-7. Теперь наконец-то мы можем создать наш е первое приложение и посмотреть как оно работает. *Я же покажу, как создать простой проект.* В Eclipse идем в меню File > New > Project. Выбираем тип проекта - Android > Android Application Project, жмём Next. Появилось окно создания проекта. Давайте заполнять. Начнём с Project Name – это имя проекта, которое будет видно в общем списке проектов слева. Т.к. проектов у нас будет много, предлагаю придумать префикс для названий. Так будет проще к ним возвращаться, и они будут красиво и удобно структурированы. Префикс может быть таким: P<номер урока(000)><номер проекта в уроке(0)> . На номер урока выделим три цифры, на номер проекта – одну. Добавим к префиксу некое смысловое название (OurFirstProject) и получим имя наш его первого проекта в третьем уроке - P0031_OurFirstProject. Application name – непосредственно имя программы, которое будет отображаться в списке приложений в смартфоне.

Можно брать имя проекта без префикса Package name – это понятие из Java, подробно можно посмотреть здесь. Вкратце – это префикс для имени классов наш его приложения. Я буду использовать ru.startandroid.develop.<имя приложения>

Build SDK определяет, возможности какой версии Android будет использовать приложение. Выберем ту же, что и при создании AVD – т.е. 4.2.2.

Minimum Required SDK определяет минимальную версию Android, на которой запустится приложение. Можно сделать ее равной Build SDK - 4.2.2.

Из галок оставляете только Create Project in Program. Проект будет создан и сохранен в дефолтном Program. Жмем Next. Открывается экран создания Activity. Здесь выбираем BlankActivity и жмем Next. Здесь убедимся, что в поле Activity Name указано MainActivity, а в поле Layout Name укажем main. Остальные поля не трогаем. Жмём Finish. Проект создан. Дальнейшие скрыни для разных версий могут отличаться, но незначительно. Слева появился наш проект, давайте раскроем его. Разберем наиболее важные и часто используемые папки и файлы:

src – все, написанные нами исходные коды, будут в этой папке

gen – генерируемые средой файлы, необходимые для приложения. Здесь лучше ничего не трогать . (Если этой папки нет - что-нибудь измените в проекте и нажмите кнопку сохранить).

Android 4.2.2 – необходимые для приложения Android-библиотеки assets и res – папки для файлов-ресурсов различного типа AndroidManifest.xml – манифест или конфиг-файл приложения. Все это мы будем в дальнейшем использовать, и станет понятнее, что и зачем нужно. Давайте выделим имя проекта слева:

и запустим его - нажмем CTRL+F11, далее выбираем Android Application,

жмем OK. Запустится эмулятор, советую в это время ничего особо не трогать и не нажимать, т.к. это глючная и капризная штука. Время запуска - около минуты. Дожидаемся, пока в консоли Eclipse появятся подобные строки.

Приложение закачано на эмулятор, установлено и запущено. Снимаем блокировку экрана в эмуляторе (если она есть) и наблюдаем наше приложение:

Видно название приложения и название Activity. Мы создали и запустили первое приложение.

Если не получилось запустить и Eclipse вывел в консоль подобное сообщение: "emulator-5554 disconnected! Cancelling'ru. startandroid. develop. OurFirstProject. MainAct activity launch'!" - то закройте эмулятор, и попробуйте снова. Если снова не получилось. перезапустите Eclipse. Если и сейчас не работает, удалите AVD и создайте новый . В итоге должно заработать, пусть и не с первой попытки. Главное - после запуска приложения (CTRL+F11) старайтесь совершать как можно меньше движений на компьютере. Я заметил четкую тенденцию - если во время запуска переключаться между различными окнами, то эмулятор запускается неправильно. А если просто посидеть и подождать минутку - то все будет хорошо. Ну и надо чтоб при этом не было включено какое-нить кодирование видео или прочие, нагружающие систему процедуры. Если у вас имя пользователя русскими буквами, то будут небольшие проблемы. На следующем лабораторном будем добавлять в наш е приложение различные элементы, и менять их свойства.

Варианты заданий к лабораторной работе №7

1. Составить программу нахождения наибольшего и наименьшего их трёх произвольных чисел.
2. Даны три числа A,B,C. Если все числа положительные вычислить Z = A+B+C, если все отрицательные - Z = (A+B)*C, в противном случае Z=A*B*C
3. Составить программу , печатающую одно из сообщений: “ Корни действительные и равные”, "Корни действительные и различные", "Корни мнимые", в зависимости от вида корней уравнения $ax^2 + bx + c = 0$ Принять a,b,c≠0

Задание к работе и порядок выполнения

- Составить программу для заданного варианта;
- Записать программу на языке AVD +JAVA;
- Отладить программу;
- Продемонстрировать работу программы на ПК для заданного
- варианта.

Домашняя подготовка. Для допуска к лабораторной работе :

- Изучить (по литературе) тему – AVD +JAVA.

Форма отчёта и порядок защиты. Студент должен :

- Представить отчёт о проделанной работе.
- Выполнить дополнительный вариант задания по указанию преподавателя.

В этой лабораторной работе мы научились разрабатывать программу на языке Java в среде ANDROID. В следующем лабораторном мы научимся, работать среде эмулятора сотового телефона.

Список использованных источников

1. Голошапов А.Л. Google Android: Программирование для мобильных устройств. СПб.: БХВ - Петербург. 2011- 448 с.
2. Горнаков С.Н. Программирование мобильных телефонов на Java 2 Micro Edition. – М. ДМК Пресс, 2004.- 336 с.
3. Хемраджани Анил. Гибкая разработка приложений на Java с помощью Spring, Hibernate и Eclipse.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2008. -352 с.

Лабораторная работа № 8

Тема работы. Эмулятор ANDROID сотового телефона.

Цель работы. Разработка программы в среде эмулятора сотового телефона ANDROID.

Пример: Вывод сообщения «Hello World» среде эмулятора сотового телефона ANDROID ниже приведено код программы Java и XML.

```
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;

public class MainActivity extends Activity {

@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
}

}
```

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context=".MainActivity" >
```

```
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/user_name" />

</RelativeLayout>
```

В процессе работы эта программа Выводит сообщение Hello World эмулятора сотового телефона , смотрите на рис.8.1.

Рис.8.1. Результат выполнения программы

Пример: Ввод кода в строку ID эмулятора сотового телефона. ниже приведено код программы Java и XML.

```
package com.example.aa;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;

public class MainActivity extends Activity {

        @Override

protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
        }

        @Override

        public boolean onCreateOptionsMenu(Menu menu) {
                // Inflate the menu; this adds items to the action bar if it is present.
                getMenuInflater().inflate(R.menu.main, menu);
                return true;
        }

}

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
     >
     <TextView
     android:layout_width="wrap_content"
          android:layout_height="wrap_content"
     android:text = "Hello!"
     />

<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Click!"
    />

<EditText
          android:layout_width = "250dp"
          android:layout_height="wrap_content"
     android:hint="Kylych"
     />

</LinearLayout>
```

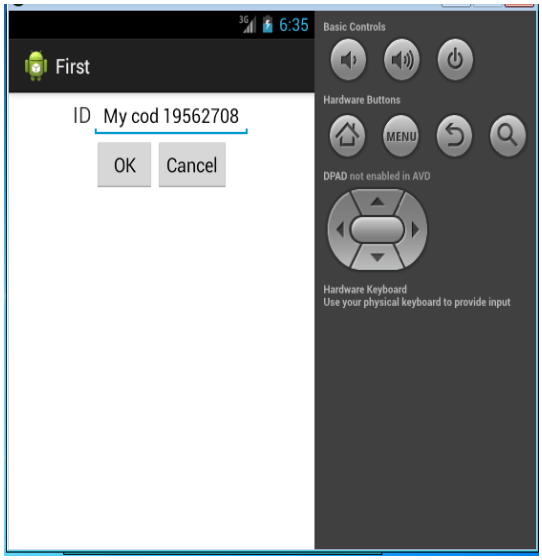


Рис.8.2. Результат выполнения программы

В процессе работы эта программа выводит следующие строки ID и OK Cancel эмулятора сотового телефона, смотрите на рис.8.2. Введен текст “Mycod 19562708 “ в строку ID.

Варианты заданий к лабораторной работе №8

1. Составить программу калькулятор.
2. Составить программу связь с интернетом.

Задание к работе и порядок выполнения

- Составить программу для заданного варианта;
- Записать программу на языке XML +JAVA;
- Отладить программу;
- Продемонстрировать работу программы на ПК для заданного
- варианта.

Домашняя подготовка. Для допуска к лабораторной работе :

- Изучить (по литературе) тему – XML +JAVA.

Форма отчёта и порядок защиты. Студент должен :

- Представить отчёт о проделанной работе.
- Выполнить дополнительный вариант задания по указанию преподавателя.

В этой лабораторной работе мы научились разрабатывать программу на языке Java , XML в среде ANDROID.

Список использованных источников

1. Голошапов А.Л. Google Android: Программирование для мобильных устройств. СПб.: БХВ - Петербург. 2011- 448 с.
2. Горнаков С.Н. Программирование мобильных телефонов на Java 2 Micro Edition. – М. ДМК Пресс, 2004.- 336 с.
3. Хемраджани Анил. Гибкая разработка приложений на Java с помощью Spring, Hibernate и Eclipse.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2008. -352 с.

Корректор ***Эркинбек к. Ж.***
Редактор ***Турдукулова А.К.***
Тех.редактор ***Кочоров А.Д***

Подписано к печати 10.04.2015 г. Формат бумаги 60x84 $^{1}/_{16}$.
Бумага офс. Печать офс. Объем 2,25 п.л. Тираж 50 экз. Заказ 199. Цена38,5 с.

Бишкек, ул. Сухомлинова, 20. ИЦ “Текник” КГТУ им. И.Раззакова, т.: 54-29-43
e-mail: beknur@mail.ru