Практическая работа №1. Знакомство со средой разработки. Синтаксис и основные управляющие конструкции языка Джава.

Цель: введение в разработку программ на языке программирования Джава.

Теоретические сведения

Язык Джава— это объектно-ориентированный язык программирования, с со строгой инкапсуляцией и типизацией. Программы, написанные на языке, Джава могут выполняться под управлением различных операционных системах при наличии необходимого ПО – Java Runtime Environment.

Для того чтобы создать и запускать программы на языке Джава необходимо следующее ПО:

- Java Development Kit (JDK);
- Java Runtime Environment (JRE);
- Среда разработки. Например, IDE IntelliJ IDEA или NetBeans

Установка необходимого для разработки на Джава программного обеспчения

Для того, чтобы скачать среду разработки, можно воспользоваться следующими ссылками:

1. Программа IntelliJ IDEA это профессиональная интегрированная среда разработки для разработки программ на Джава: https://www.jetbrains.com/idea/download/#section=windows
2. Программа “NetBeans IDE”: https://netbeans.org/downloads/
3. JDK: http://www.oracle.com/technetwork/java/javase/downloads/index.html

По умолчанию, скаченный JDK установится в папку с таким адресом:C:\Program Files\Java на компьютере с ОС Windows

Начало работы с программой IntelliJ IDEA

После установки одной из сред разработки (“IntelliJ IDEA” или “NetBeans IDE”) можно начать создавать проекты. Далее будет показано, как начать новый проект на примере программы «IntelliJ IDEA». Рассмотрим подробно, по шагам процесс создания:

1. В открытом окне программы выбираем “Create New Project”.

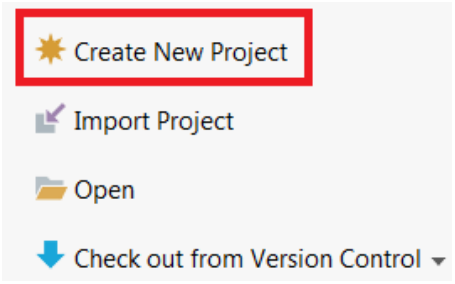

Рисунок 1.1 – Пункт в меню для выбора для создания рпоекта

2. Щёлкаем «New», чтобы загрузить JDK.

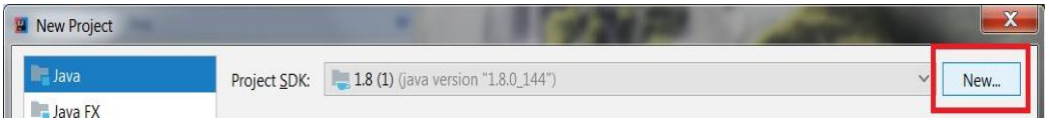

Рисунок 1.2 - Поле – путь к SDK(скачали ранее)

В этом поле настраивается путь к SDK, котрый вы предварительно скачали с сайта oracle.

3. Из выпадающего списка папок выбираем «Program Files».

Рисунок 1.3 – Пункт выбора папки с SDK

4. В «Program Files» выбираем папку «Java».

Рисунок 1.4 – Папка с SDK

5. Далее выбираем папку «jdk…».

Рисунок 1.5 – Выбираем папку с jdk

6. Затем дважды нажимаем «Next» в нижнем правом углу.

Рисунок 1.6 – Подтверждение выбора

Выбираем название для будущего проекта и затем нажимаем кнопку «Finish».

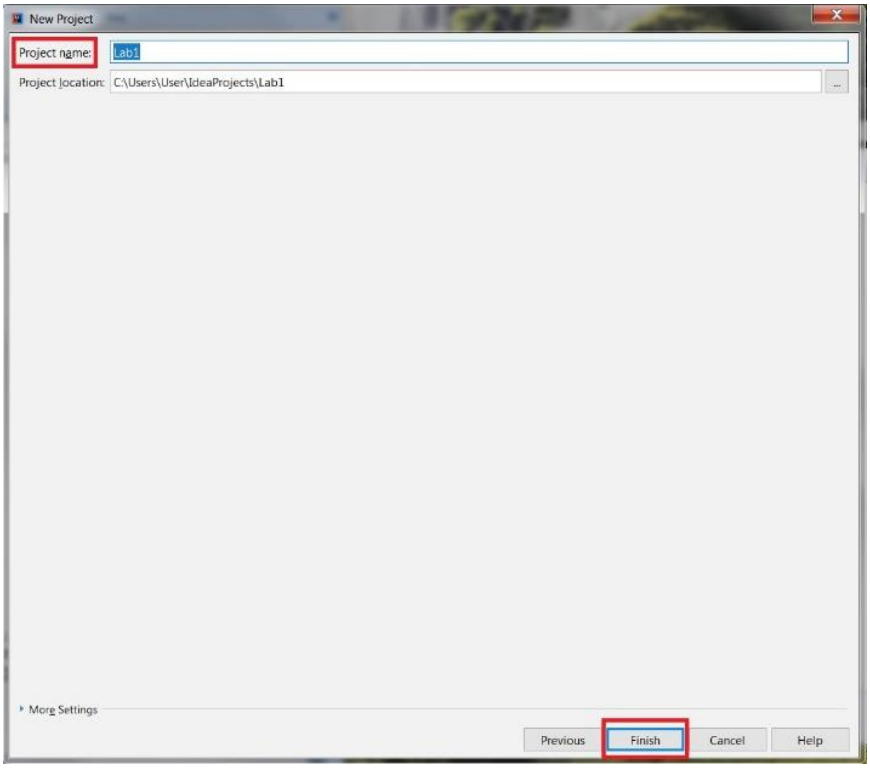

Рисунок 1.6 – Поле для ввода названия проекта

7. Щёлкаем правой кнопкой мыши по папке «src» в дереве файлов проекта и создаем новый пакет.

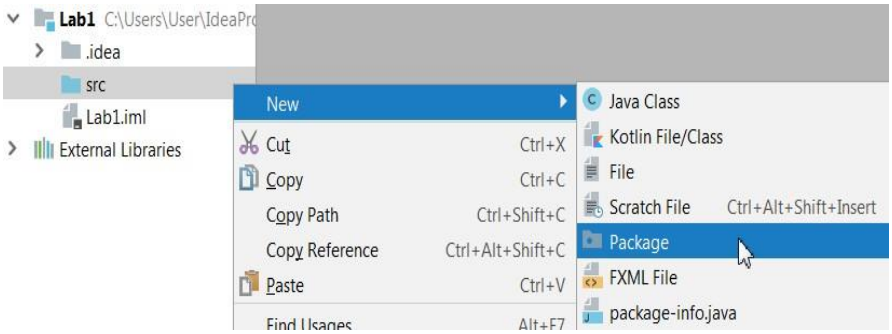

Рисунок 1.7 – Папка с исходным кодом

8. Вводим название пакета. «Package» – это оператор, который сообщает транслятору, в каком пакете должны определяться содержащиеся в данномфайле классы.

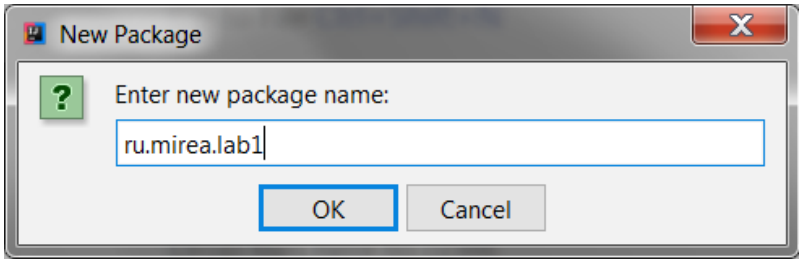

Рисунок 1.8 – Поле для ввода названия пакета

Щелкаем по созданному пакету правой кнопкой мыши и создаем новыйкласс.

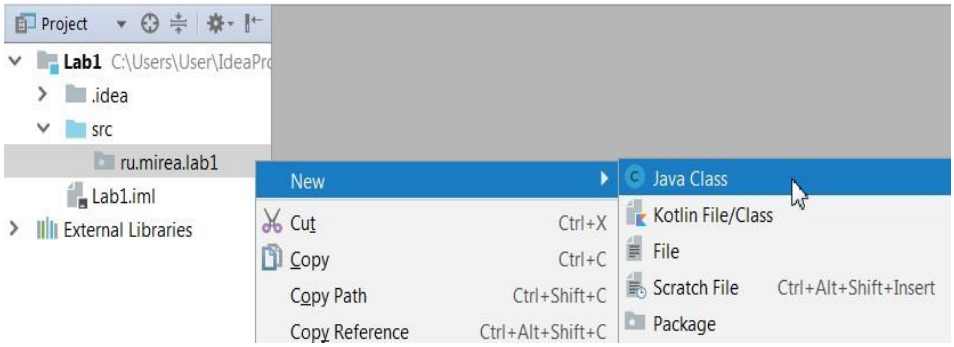

Рисунок 1.9 – Меню выбора для создания класса

9. Новый проект создан. Теперь можно приступать к написанию кода.

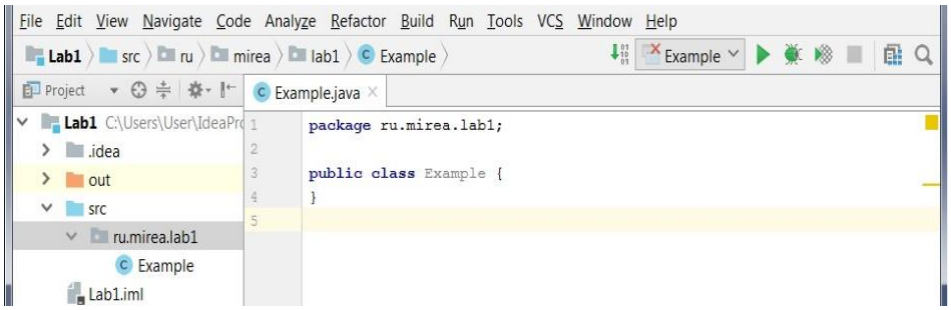

Рисунок 1.10 – Окно проекта и файл класса

Чтобы начать написание программы необходимо запустить среду разработки. При первом запуске среды обычно нужно указать путь к JDK, чтобы можно было компилировать код и запускать программу. В среде разработки необходимо создать Джава проект, после чего необходимо создать пакет и в нем создать какой-либо класс. По правилам именования пакетов используются только строчные буквы, в названии класса первая буква имени должна быть заглавная. Также в свойствах проекта нужно указать класс, с которого будет начинаться запуск программы.

В классе, с которого будет начинаться запуск программы обязательно должен быть статический метод main(String[]), который принимает в

качестве аргументов массив строк и не возвращает никакого значения.

Листинг 1.1 Пример объявления класса:

```
package example;
public class Example {
public static void main(String[] args) {
}}
```

В этом примере создан класс Example, располагающийся в пакете example. В нем содержится статический (ключевое слово static) метод main. Массив строк, который передается методу main() - это аргументы командной строки. При запуске Джава программы, выполнение начнется с метода main().

Переменные класса.

Чтобы объявить переменную, необходимо указать тип переменной и ее имя. Тип переменной может быть разным: целочисленный (long, int,short, byte), число с плавающей запятой (double, float), логический(boolean), перечисление, объектный (Object).

Переменным можно присваивать различные значения с помощью оператора присваивания “=”.

Целочисленным переменным можно присваивать только целыечисла, а числам с плавающей запятой - дробные. Целые числа обозначаются цифрами от 0 до 9, а дробные можно записывать отделяюцелую часть от дробной с помощью точки. Переменным типа float необходимо приписывать справа букву “f”, обозначающую, что данное число типа float. Без этой буквы число будет иметь тип double.

Класс String - особый класс в языке Джава, так как ему можно присваиватьзначение, не создавая экземпляра класса (Джава это сделает автоматически). Этот класс предназначен для представления строк. Строковое значение записывается буквами внутри двойных кавычек.

Листинг 1.2 Пример присваивания значений переменным

```
float length = 2.5f;
double radius = 10024.5;
int meanOfLife = 42;
Object object = new String("Hello, world!");
String b = "Once compiled, runs everywhere?";
```

С целочисленными переменными можно совершать различные операции: сложение, вычитание, умножение, целое от деления, остатокот

деления. Эти операции обозначаются соответственно “+”, “-”, “*”, “/”,“%”. Для чисел с плавающей запятой применимы операции сложения, вычитания, умножения, деления. Для строк применима операция “+”, обозначающая конкатенацию, или слияние строк.

Массивы в Джава

Массив — это конечная последовательность элементов одного типа, доступ к каждому элементу в которой осуществляется по его индексу.

Для того чтобы создать массив переменных, необходимо указать квадратные скобки при объявлении переменной массива. После чего необходимо создать массив с помощью оператора new. Необходимо указать в квадратных скобках справа размер массива. Например, чтобы создать массив из десяти целочисленных переменных типа int, можно написать так:

```
int[] b = new int[10];
```

Для того чтобы узнать длину массива, необходимо обратиться к его свойству length через точку, например b.length для нашего прримера объявления массива.

Для того чтобы получить какой-либо элемент массива, нужно после имени массива в квадратных скобках индекс, или порядковый номер элемента. Нумеруются в массивах начинается с нуля, как и в языке С.С++. Например, чтобы получить 5 элемент массива, можно написать так: b[4].

Условные конструкции в Джава

Условие — это конструкция, позволяющая выполнять то или другое действие, в зависимости от логического значения, указанного в условии. Синтаксис создания условия на Джава следующий:

```
if(a==b) {
//Если a равно b, то будут выполняться операторы в этой области
} else {
//Если a не равно b, то будут выполняться операторы в этой области
}
```

Если логическое условие, указанное в скобках после ключевого

слова if, истинно, то будет выполняться блок кода, следующий за if, иначе будет выполняться код, следующий за ключевым словом else. Блок else не обязателен и может отсутствовать.

Скобками “{”, “}” обозначается блок кода, который будет

выполняться. Если в этом блоке всего 1 оператор, то скобки можно не писать (для условий и циклов).

Логическое условие составляется с помощью переменных и операторов равенства, неравенства, больше, меньше, больше или равно, меньше или равно, унарная операция не. Эти операторы обозначаются соответственно “==”, “!=”, “>”, “<”, “>=”, “<=”, “!”. Результатом сравнения является логическое значение типа boolean, которое может иметь значение true (“истина”) или false (“ложь”). Логические значения могут храниться в переменных типа boolean.

Циклические конструкции в Джава

Цикл — это конструкция, позволяющая выполнять определенную часть кода несколько раз. В Джава есть три типа циклов for, while, do while. Цикл for — это цикл со счетчиком, обычно используется, когда известно, сколько раз должна выполниться определенная часть кода.

Итерационный цикл for

Синтаксис цикла for:

```
for(int i=0;i<10;i++) {
//Действия в цикле
}
```

В данном примере, в цикле объявлена переменная i, равная изначально 0. После точки с запятой “;” написано условие, при котором будет выполняться тело цикла (пока i<10), после второй точки с запятой указывается как будет изменяться переменная i (увеличиваться на единицу каждый раз с помощью операции инкремента “++”). Прописывать условие, объявлять переменную и указывать изменение переменной вцикле for не обязательно, но обязательно должны быть точки с запятой.

Цикл с предусловием while

Цикл while — это такой цикл, который будет выполняться, пока логическое выражение, указанное в скобках истинно.

Синтаксис цикла while:

```
while(logic) {
//Тело цикла
}
```

В данном примере тело цикла будет выполняться, пока значение логической переменной logic равно true, то есть истинно.

Цикл с постусловием do while

Цикл вида do while — это такой цикл, тело которого выполнится хотя быодин раз. Тело выполнится более одного раза, если условие, указанное в скобках истинно.

```
do {
//Тело цикла
}while(logic);
```

Потоки ввода/вывода и строки в Java, класс String

Для ввода данных используется класс Scanner из пакета java.util.

Этот класс надо импортировать в те программе, где он будет использоваться. Это делается до начала определения класса в коде программы.

В классе есть методы для чтения очередного символа заданного типа со стандартного потока ввода, а также для проверки существования такого символа.

Для работы с потоком ввода необходимо создать объект класса Scanner, при создании указав, с каким потоком ввода он будет связан. Стандартный поток ввода (клавиатура) в Джава представлен объектом — System.in. А стандартный поток вывода (дисплей) — уже знакомым вам объектом System.out. Есть ещё стандартный поток для вывода ошибок — System.err

Листинг 1.3 – Пример чтения данных ввода с клавиаьуры

```
import java.util.Scanner; // импортируем  класс
import java.util.Scanner; // импортируем  класс
public class Main {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in); /
/ создаёмобъект класса  Scanner
int i = 2; System.out.print("Введитецелоечисло:
");if(sc.hasNextInt()) {
/* hasNextInt()возвращает истинну если с потока
ввода можно считать целое число*/
i = sc.nextInt(); /* считывает целое число из
потока ввода и сохраняет в переменную i*/
System.out.println(i*2);
```

```
} else {
System.out.println("Вы ввели не целое число");
}}
```

В Джава имеется также метод nextLine(), позволяющий считывать целую последовательность символов, т.е. строку, а, значит, полученное через этот метод значение нужно сохранять в объекте класса String. В следующем примере создаётся два таких объекта, потом в них поочерёдно записывается ввод пользователя, а далее на экран выводится одна строка, полученная объединением введённых последовательностей символов.

Листинг 1.4 – Пример чтения строк и их конкатенации

```
import java.util.Scanner;
public class Main {
public static void main(String[]args) {
Scanner sc = new Scanner(System.in);
String s1,s2;
s1 = sc.nextLine();
s2 = sc.nextLine();
System.out.println(s1 + s2);
}}
```

Существует и метод hasNext(), проверяющий остались ли впотоке ввода какие-то символы.

В классе String существует масса полезных методов, которые можно применять к строкам (перед именем метода будем указывать тип того значения, которое он возвращает). Некотрые из метолов:

- intlength() — возвращает длину строки (количество символов в ней);
- boolean isEmpty() — проверяет, пустая листрока;
- String replace(a, b) — возвращает строку, где символ a (литералили переменная типа char) заменён на символb;
- String toLowerCase() — возвращает строку, где все символы исходной строки преобразованы кстрочным;
- String toUpperCase() — возвращает строку, где все символы исходной строки преобразованы кпрописным;
- boolean equals(s) — возвращает истинну, если строка к

которой применён метод, совпадает со строкой s указанной в аргументе метода (с помощью оператора == строки сравнивать нельзя, как и любые другиеобъекты);

- int indexOf(ch) — возвращает индекс символа ch в строке (индекс это порядковый номер символа, но нумероваться символы начинают с нуля). Если символ совсем не будет найден, то возвратит -1. Если символ встречается в строке несколько раз, то вовзвратит индекс его первоговхождения.
- int lastIndexOf(ch) — аналогичен предыдущему методу, но возвращает индекс последнего вхождения, если смивол встретился в строке несколькораз.
- int indexOf(ch,n) — возвращает индекс символа ch в строке, но начинает проверку с индекса n (индекс это порядковый номер символа, но нумероваться символы начинают с нуля). char charAt(n) — возвращает код символа, находящегося в строке под индексом n (индекс это порядковый номер символа, но нумероваться символы начинают с нуля).

Листинг 1.5 – Пример работы с методами String

```
public class Main {
public static void main(String[] args) {
String s1 = "firefox";
System.out.println(s1.toUpperCase());
// выведет «FIREFOX»
String s2 = s1.replace('o', 'a');
System.out.println(s2); // выведет «firefax»
System.out.println(s2.charAt(1)); // выведет «i»
int i;
i = s1.length();
System.out.println(i);
i = s1.indexOf('f');
System.out.println(i);
i= s1.indexOf('r');
System.out.println(i);
i =s1.lastIndexOf('f');
System.out.println(i);
i = s1.indexOf('t');
```

```
System.out.println(i);
i = s1.indexOf('r',3);
System.out.println(i);
}}
```

Пример программы, которая выведет на экран индексы всех пробелов в строке, введенноё пользователем с клавиатуры:

Листинг 1.6 – Пример работы со строками

```
importjava.util.Scanner;
public class Main {
public static void main(String[] args) {
Scanner sc = new
Scanner(System.in);
String s =sc.nextLine();
for(int i=0; i < s.length(); i++) {
   if(s.charAt(i) == ' '){
    System.out.println(i);
   }
}
}}
```

Методы в языке Java.

Методы позволяют выполнять блок кода, из любого другого места, где это доступно. Методы определяются внутри классов. Методы могут быть статическими (можно выполнять без создания экземпляра класса), не статическими (не могут выполняться без создания экземпляра класса). Методы могут быть открытыми(public), закрытыми(private). Закрытые методы могут вызываться только внутри того класса, в котором они определены. Открытые методы можно вызывать для объекта внутри других классов.

При определении метода можно указать модификатор доступа (public, private, protected), а также указать статический ли методключевым словом static. Нужно обязательно указать тип возвращаемогозначения и имя метода. В скобках можно указать аргументы, которые необходимо передать методу для его вызова. В методе с непустым типом возвращаемого значения нужно обязательно указать оператор return и значение, которое он возвращает. Если метод не возвращает никакого

значения, то указывается тип void.

Пример определения метода класса:

```
public static int sum(int a, int b) {
return a+b;
}
```

В данном примере определен метод, возвращающий сумму двух чисел a и b. Этот метод статический, и его можно вызывать, не создавая экземпляра класса, в котором он определен.

Чтобы вызвать этот метод внутри класса, в котором он создан необходимо написать имя метода и передать ему аргументы. Пример:

```
int s = sum(10,15);
```

Задание на практическую работу №1

1. Создать проект в IntelliJ IDEA
2. Создать свой собственный Git репозитарий
3. Написать программу, в результате которой массив чисел создается с помощью инициализации (как в Си) вводится и считается в цикле сумма элементов целочисленного массива, а также среднее арифметическое его элементов результат выводится на экран. Использовать цикл for.
4. Написать программу, в результате которой массив чисел вводится пользователем с клавиатуры считается сумма элементов целочисленного массива с помощью циклов do while, while, также необходимо найти максимальный и минимальный элемент в массиве, результат выводится на экран.
5. Написать программу, в результате которой выводятся на экран аргументы командной строки в цикле for.
6. Написать программу, в результате работы которой выводятся на экран первые 10 чисел гармонического ряда (форматировать вывод).
7. Написать программу, которая с помощью метода класса, вычисляет факториал числа (использовать управляющую конструкцию цикла), проверить работу метода.
8. Результаты выполнения практической работы залить через IDE в свой репозитарий и продемонстрировать преподавателю.