Занятие 1.

1. Eclipse – среда разработки, предназначенная для java-разработки.

Для того, чтобы создать программу, необходимо создать проект. В проекте есть папка src. В эту папку нужно помещать исходный код. С целью предотвращения одинакового именования классов следует создавать packages. Package – структура папок в операционной системе. Принято именовать следуюшим образом: первый уровень – тип организации (com, edu, by и тд), второй уровень – название организации (bigroi, iba, epam, apple, google и тд), третий уровень – название проекта (calculator, jurnal и тд), четвертый и далее – группы классов в проекте. Имя package принято писать в нижнем регистре.

В package помещаются классы. Классы хранят исходный код. Класс – текстовый файл в операционной системе. В классе создаются методы. Метод – набор инструкций (операторов), которые выполняются последовательно. Метод

Public static void main (String [] ) {} является точкой входа в программу, т.е. метод, который будет выполнен первым при запуске. Метод состоит из заголовка и тела. Все, что до {} скобок является заголовком, в {} – телом метода. Инструкции пишутся в теле. **В теле метода нельзя писать другой метод.**

1. Программа. Основной целью программы является обработка данных. Данных хранятся в переменных. Переменные имеют тип. В java есть две разновидности типов: примитивный и ссылочный. Примитивных типов 8:

* int – целое число, от -2 млрд до +2 млрд
* long – целое число, > int
* short – целое число, <int
* byte – целое число, <short (-128 до 127)
* double – дробное число,
* float – дробное число, < double
* char – символ, например (a,b,c, 1,3,-,^ и тд)
* boolean – логический тип, принимающий значения true/false
* void – тип данных, обозначающих что данных нет. Переменные типа void заводить нельзя.

Целочисленные константы по умолчанию имеют тип int. Дробные константы имеют тип double.

Чтобы завести переменную, необходимо в теле метода написать тип переменной и ее имя. Имя должно быть существительным с маленькой буквы. Если много слов, то последующее слово пишется с большой буквы без пробела.

Например,

int count = 10;

double averageMark = 9.3;

1. Операторы.

* Присвоение. Обозначается «=». Помещает в переменную слева значение справа. count = 20; averageMark = (10.0+8.0)/2.0; Нельзя присваивать значение большего типа меньшему типу. float f = 3,14; // ошибка.
* Арифметические действия. «+», «-», «\*», «/», «%». Оператор деления может быть целочисленного деления или дробного деления. Целочисленное деление означает целую часть от деления. count = 5/3; // count 🡪 1.

Целочисленное деление работают когда оба операнда целые числа. Если одно из чисел дробное, то осуществляется дробное деление. averageMark = 5/2.0 // averageMark 🡪 2,5 ;

Остаток от деления(%). Оба операнда должны быть целыми числами. count = 5/3; // count 🡪 2.

* Операторы сравнения. «<» «>» «<=» «>=» «==» «!=». Сравнивают два числа, результат – boolean. Если числа например равны (используем оператор «==»), вернется true. Иначе – false.
* Логические операторы. «&» «|» «^» «!». Операнды должны быть типа boolean. Таблица истинности:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | & |  | a | b | | |  | a | b | ^ |  | a | !a |  |
| false | false | false |  | false | false | false |  | false | false | false |  | false | true |  |
| False | true | false |  | false | true | true |  | false | true | true |  | true | false |  |
| true | false | false |  | true | false | true |  | true | false | true |  |  |  |  |
| true | true | true |  | true | true | true |  | true | true | false |  |  |  |  |

Удвоение && обозначает, что результат вернется false, если первый операнд равен false. При этом значение второго операнда читаться не будет.

int i = 0;

i != 0 && (10/i) > 0; // если бы & - была бы ошибка деления на 0 .

Удваение || обозначает что результат вернется true, если первый операнд равен true. При этом значение второго операнда читаться не будет.

* Оператор условного перехода if. Если значение в скобках истина, то выполняется код в фигурных скобках. Иначе код опускается.

int i = 0;

if (i>0){

System.out.println(i); // код не выполнится.

}

* Оператор условного перехода if - else. Если значение в скобках истина, то выполняется код в фигурных скобках под if. Иначе выполняется код в фигурных скобках под else. опускается.

int i = 0;

if (i>0){

System.out.println(i); // код не выполнится.

} else {

System.out.println(“Hello”); // код выполнится.

}

* Оператор условного перехода switch. Ищет среди лейблов после ключевого слова case значение равное аргументу. И передает управление на эту строку. Все строки ниже case будут выполнены. Если лейбл не найден, управление переходит на строку с ключевым словом default. Если default нет, то switch пропускается.

int i = 20;

switch (i){

case 0: System.out.println(0); // код не выполнится

case 1: System.out.println(1); // код не выполнится

case 20: System.out.println(20); // код выполнится

case 2987: System.out.println(2987); // код выполнится

default: System.out.println(“Hello”); // код выполнится

}

Для того, чтобы выполнилось только нужный case, но не нижестоящие, используйте ключевое слово break.

Оператор break передает управление следующей инструкции после switch или цикла.

* Оператор цикла с предусловием. Цикл может выполнять свое тело многократно. Цикл с предусловием сначала проверяет равен ли true аргумент, и затем выполняет тело.

int i = 0;

while (i<10){

i++;

System.out.println(i);

} // в консоль выведутся числа от 1 до 10.

* Оператор цикла с постусловием. Цикл с постусловием сначала выполняет свое тело, а затем проверяет равен ли true аргумент.

int i = 0;

do {

i++;

System.out.println(i);

} while (i<10); // в консоль выведутся числа от 1 до 10.

* Оператор цикла счетчик. Цикл с предусловием, который помимо условия имеет код инициализации цикла и код инициализиции итерации.

for (1;2;3){

} // 1- инициализация цикла. Выполняется только 1 раз, при первом входе в цикл. 2 – условие. Если true – тело цикла выполняется. 3- Инициализация итерации. Выполняется перед каждой проверкой условия, начиная со второго раза.

for (int i = 0; i<10; i++){

System.out.println(i);

} // в консоль выведутся числа от 0 до 9.