ЛАБОРАТОРНАЯ РАБОТА №2. ТИПЫ ДАННЫХ И УПРАВЛЯЮЩИЕ СТРУКТУРЫ JAVA

Цель работы: получить практические навыки работы с управляющими структурами на языке Java.

Примитивные типы данных:

− byte;

− short;

− int;

− long;

− float;

− double;

− char;

− boolean.

Ключевые слова выполнения цикла:

− do;

− while;

− for;

− break;

− continue.

Ключевые слова, выполняющие переход:

− if;

− else;

− switch;

− case;

− default;

− break.

Метод, переменная и модификаторы класса:

− private;

− public;

− protected;

− final;

− static;

− abstract;

− synchronized;

− volatile;

− strictfp.

Буквальные константы:

− false;

− true;

− null.

Ключевые слова, связанные с методом:

− return;

− void.

Ключевые слова, связанные с пакетом:

− package;

− import.

Ключевые слова, связанные с обработкой особых ситуаций:

− try;

− catch;

− finally;

− throw;

− throws.

Другие ключевые слова:

− new;

− extends;

− implements;

− class;

− instanceof;

− this;

− super.

Данные ключевые слова вам необходимо знать и уметь объяснить их суть и привести примеры использования в коде, потому что Давид Дмитриевич будет всё спрашивать))))))

Основные типы данных Язык Java является строго типизированным. Это значит, что каждая переменная и каждое выражение имеют свой определенный тип. Автоматического приведения типов попросту не существует. Иногда, конечно, возникает необходимость работать с переменными разных типов, и Java позволяет это сделать, но об этом – немного позже.

Всего в Java существует восемь простых типов данных, которые условно можно разделить на четыре группы:

целые (Integers) – описывает целые числа с учетом знака. Включает в себя byte, short, int, long;

числа с плавающей запятой (Floating-point number) – описывает «дробные» числа. Их нельзя считать полноценными дробными, т. к. они имеют определенную погрешность. Это типы: float и double;

символьный тип (Characters) – это тип char, который предназначен для описания символов (буквы, цифры, математические знаки, и др.);

логический тип (Boolean) – это тип Boolean, описывающий переменные, которые могут принимать значение «правда» (true) или «ложь» (false).



Назначение переменных и инициализация

Переменная – значение, которое может измениться по мере необходимости в течение выполнения программы; они представлены символическими именами. Другими словами, значение переменной изменяется всякий раз, когда назначается новое значение. С точки зрения программиста, переменная имеет три характеристики:

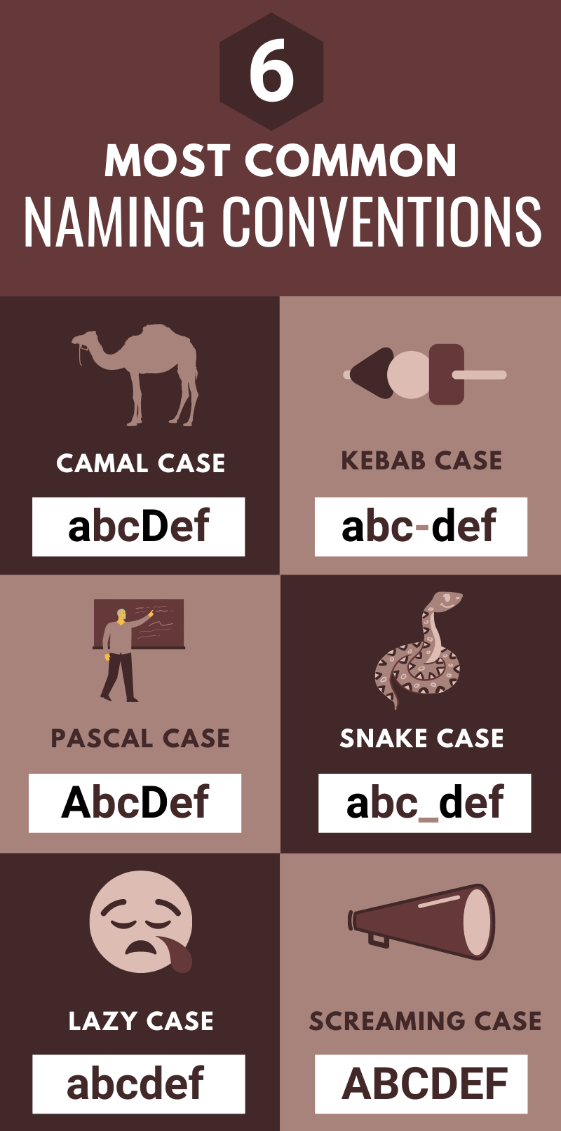
− имя;

− начальное значение;

− область видимости.

Имя переменной также называют идентификатором. Всякий раз, когда объявляется переменная, ей присваивается значение, и это значение является значением по умолчанию. Переменные также имеют свою область, которая определяет их видимость и время жизни в различных местах программы.

Один из важных моментов названия любых переменных – правильное наименование. Всего есть несколько вариантов:



Каждый из вариантов используется при наименовании разных переменных, констант, полей, объектов, классов и т.д.

Мы предлагаем самостоятельно рассмотреть, где какой случай используется!

Пакет java.lang содержит классы, которые являются базовыми в языке Java. Этот пакет содержит классы, которые являются фундаментальными для дизайна языка программирования Java. Самые важные классы – Object, который является корнем иерархии класса, и Class, образцы которого представляют классы во время выполнения.

Часто необходимо представить значение примитивного типа, как будто это объект. Обертка классов Boolean, Character, Integer, Long, Float и Double реализует эту цель. Объект типа Double, например, содержит поле, тип которого является double, представляя значение таким способом, что ссылка к этому полю может быть сохранена в переменной типа ссылки. Эти классы также обеспечивают множество методов для того, чтобы преобразовать среди примитивных значений, также поддерживать такие стандартные методы подобно «равняется» и hashCode. Void класс – noninstantiable класс, который содержит ссылку на объект Class, представляющий примитивный тип void.

Класс Math обеспечивает математические функции типа синуса, косинуса, и квадратного корня. Классы String и StringBuffer, подобно классу Math, обеспечивают обычно используемые операции на символьных строках.

Классы ClassLoader, Process, Runtime, SecurityManager и System обеспечивают «системные операции», которые управляют динамической загрузкой классов, созданием внешних процессов, ведущих запросов среды типа времени дня и принуждения политики безопасности.

Некоторые статические методы доступны в java.lang. Класс Math – чтобы исполнять математические операции. Статические методы – методы, к которым можно непосредственно обратиться с помощью класса, не создавая объектную ссылку класса.

Компьютер хорошо умеет исполнять повторные операции. Он может исполнять такие операции неустанно 10, 100 или даже 10 000 раз. Каждый машинный язык должен иметь особенности, которые инструктируют компьютер, как исполнять такие повторные задачи, и эти задачи решаются с помощью управляющих структур.

Управляющие структуры Java:

− lf-else;

− switch;

− loops:

− while;

− do-while;

− for.

Конструкция if-else также известна как условная управляющая структура, или структура выбора, потому что она проверяет данное условие и исполняет указанную задачу в зависимости от того, является ли условие истинным или ложным.

Объявление switch используется, когда должны быть выполнены многократные сравнения состояний. Оператор switch может также заменять длинный ряд вложенных операторов if-else-if. Оператор switch может объявляться выражением или переменной.

While loop также известен как итерация. Это наиболее широко используемая структура циклов loop. Если while loop содержит только одно утверждение, то изогнутые фигурные скобки писать не надо. Но если есть более одного утверждения, то они должны быть включены в пределы изогнутых фигурных скобок; иначе loop выполняет только первое утверждение, пока условие истинно, при этом другие утверждения просто игнорируются. Подобно if-else, условие while loop возвращает значение булевской переменной или выражение, которое возвращает булевское значение.

Конструкция do-while loop работает подобно while-loop за исключением того, что в данной конструкции цикл выполняется по крайней мере один раз, даже если условие не является истиной.

For – это компактная форма while loop которая объединяет инициализацию переменной, проверку условия и увеличения или уменьшение значения переменной для итерации в отдельном утверждении.

Работа с массивами в Java

Массив – это список значений переменных с одинаковым именем, для обращения к значениям которых используется индекс или список индексов. Как только размер массива объявлен, то он не может быть изменен. Массивы полезны, поскольку один и тот же тип данных может быть сохранен в одном месте. Данными могут быть примитивный тип данных или объект. К индивидуальным элементам массива можно обратиться с помощью имени массива и местоположения данного элемента. Местоположение также называют индексом.

Массив может быть объявлен и представлен в памяти, как показано ниже

char ch[] = new char[10].

Вышеупомянутая запись создает массив 10 символов с именем ch.

Этот массив представлен следующим образом:

1) все элементы массива ch сохранены последовательно;

2) если мы хотим обратиться к символу, сохраненному в положении 6 массива ch, и назначить ему значение символьной переменной, просто пишем: char element = ch[5];

3) если мы хотим обратиться к 5-му элементу массива ch и назначить ему значение переменной, просто пишем: char element = ch[4];

4) здесь 5 и 4 – индексы массива ch. Индекс начинается с 0. Следовательно, чтобы обратиться к 5-му символу, индекс должен быть 4.

Рассмотрим следующие различные способы создания массивов:

char ch[]; // just a declaration.

Здесь длина массива не определена.

char ch[] = new char[10]; // declaration and creation.

Ключевое слово «new» используется для распределения памяти для массива. Первая запись (выше) объявляет массив, а вторая запись объявляет и создает массив.

Создание массива означает, что выделяется фактическая память для сохранения элементов массива. Это также называется реализацией массива. Когда массив создается, всем элементам массива назначаются значения по умолчанию в зависимости от типа массива. Если массив типа int, то всем элементам массива по умолчанию присваивается нуль. Если массив имеет переменную типа boolean, то все элементы имеют значение false. Это называется автоинициализация.

Действующая инициализация В этом типе элементы массива инициализированы во время объявления:

int a[] = {10,20,30,40,50};

float floatArray[] = {248.75,45.50,873.45};

Задания для самостоятельной работы.

Задание: написать программу, заполняющую целочисленный вектор (массив) случайными значениями в диапазоне от 100 до 200. Размер вектора 30 элементов. Вывести получившийся вектор на экран. Выполнить сортировку вектора по возрастанию. Вывести результат сортировки на экран.

Реализуйте метод fillRandomIntVector(), на вход которому подается размер вектора, максимальное и минимальное значения, которые могут принимать элементы вектора. Метод возвращает вектор, заполненный случайными целочисленными значениями.

Доработайте метод printVector(). Добавьте к нему параметром String message сообщение, которое выводится перед распечаткой элементов одномерного целочисленного массива. Пример вызова:

Задания по вариантам.

I. Определить одномерный массив и заполнить его случайными значениями от -99 до 99.   
Добавить метод CreateReciprocalFraction, который создаст новый массив, где каждый элемент станет обратным:

Добавить обработку ошибки деления на ноль через блок try-catch.

1) Составить и вывести на экран новый массив с номерами элементов исходного массива, которые равны заданному значению. Заданное значение определяется константой;

2) Поменять местами максимальный и минимальный элементы массива. Вывести измененный массив на экран;

3) Все элементы массива, меньшие заданного значения, и их номера записать в новые массивы. Вывести новые массивы на экран. Заданное значение определяется константой;

4) Определить дополнительный массив разрешенных значений. Определить и вывести на экран, сколько элементов исходного массива имеют разрешенные значения;

5) Определить дополнительный массив разрешенных значений. Составить массив из элементов исходного массива, имеющих неразрешенные значения. Вывести результативный массив на экран;

6) Составить и вывести на экран массив с N максимальными значениями исходного массива. N определяется константой;

7) Переписать элементы массива в обратном порядке на том же месте. Вывести измененный массив на экран;

8) Определить дополнительный массив, состоящий из неповторяющихся элементов исходного массива и вывести его на экран;

9) Составить и вывести на экран массив номеров элементов исходного массива, встречающихся один раз;

10) Определить дополнительный массив, состоящий из повторяющихся элементов исходного массива и вывести его на экран.

11) Вычислить сколько элементов данного массива больше своего предыдущего элемента.

12) Найти максимальный элемент в одномерном массиве х. Затем каждый элемент в массиве разделить на максимальный элемент. 13) Дан массив b (n) . Переписать в массив C(n) положительные элементы массива b(n) умноженные на 5. ( сжатие массива)

14) Дан одномерный массив а(n), в котором находится единственный нулевой элемент. Найти где он находится и вычислить сумму последующих за ним элементов. Выдать на экран номер элемента и сумму.

15) Определить сколько раз в этом массиве меняется знак. Например, в массиве 1,-34, 8, 14, -5, -8, -78 , 3 знак меняется 4 раза.

16) Найти минимальный элемент в одномерном массиве х. Затем каждый элемент в массиве умножить на минимальный элемент.

II. Действия над массивами.

1) Дан массив b (n) . Переписать в массив C(n) положительные элементы массива b(n) умноженные на 5 (со сжатием, без пустых элементов внутри) .Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

2) Дан массив а (n) .Переписать в массив b(n) только положительные элементы массива а, умноженные на 3. (со сжатием., без пустых элементов внутри) . Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

3) Дан массив х (n) . Переписать в массив y(n) отрицательные элементы массива х умноженные на 2. (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

4) Дан массив b (n) . Переписать в массив C(n) отрицательные элементы массива b(n) умноженные на 4. (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

5) Дан массив b (n) . Переписать в массив C(n) положительные элементы массива b(n) деленные на 5. (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

6) Дан массив а (n) .Переписать в массив b(n) только положительные элементы массива а, деленные на 3 (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

7) Дан массив х (n) . Переписать в массив y(n) отрицательные элементы массива х деленные на 2. (со сжатием., без пустых элементов внутри). Затем упорядочить по возрастанию новый массив.

8) Дан массив b (n) . Переписать в массив C(n) отрицательные элементы массива b(n). (со сжатием., без пустых элементов внутри) Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

9) Дан массив с (n) . Переписать в массив х (n) все ненулевые элементы массива умноженные на 4. (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

10) Дан массив с (n) . Переписать в массив х (n) все ненулевые элементы массива возведенные в квадрат. (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

11) Дан массив с (n) . Переписать в массив х (n) все ненулевые элементы массива (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

12) Дан массив с (n) .Переписать в массив х ненулевые элементы массива с разделенные на 5. (со сжатием., без пустых элементов внутри). Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

13) Дан массив b (n) . Переписать в массив C(n) корни квадратные из положительных элементов массива b(n) деленные на 5. (со сжатием., без пустых элементов внутри) Затем упорядочить методом «выбора и перестановки» по возрастанию новый массив.

14) Дан массив b (n) . Переписать в массив C(n) корни квадратные из положительных элементов массива b(n) (со сжатием., без пустых элементов внутри) Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

15) Дан массив х (n) . Переписать в массив y(n) элементы массива х, большие 3. (со сжатием., без пустых элементов внутри) . Затем упорядочить методом «выбора и перестановки»по возрастанию новый массив.

16) Дан одномерный массив а(n), в котором находится единственный нулевой элемент. Найти где он находится, и упорядочить по возрастанию элементы, расположенные за ним. Выдать на экран номер элемента и упорядоченный массив.

III. Матрицы. Определить матрицу (двумерный массив) и ее заполнить случайными значениями.

Построить вектор В, которой возвращает –

1) число неотрицательных элементов в i-й строке;

2) среднее арифметическое положительных элементов в i-м столбце;

3) минимальное значение в i-й строке;

4) максимальное значение в i-м столбце;

5) номер максимального значения в i-й строке;

6) номер минимального значения в i-м столбце;

7) число элементов i-й строки, значения которых меньше заданного значения;

8) значение элемента матрицы, не равное заданному значению;

9) равно 1, если значения элементов i-й строки упорядочены по возрастанию, и 0, в противном случае;

10) количество четных чисел в i-й строке.

11) сумму положительных элементов в каждом столбце матрицы.

12) произведение положительных элементов в каждом столбце матрицы.

13) количество положительных элементов в каждом столбце матрицы.

14) среднее арифметическое положительных элементов в каждом столбце матрицы

15) среднее геометрическое положительных элементов в каждом столбце матрицы

16) сумму отрицательных элементов в каждом столбце матрицы.

IV. Действия над матрицами.

1. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти номер минимального элемента её побочной диагонали.

2. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти сумму номеров минимального и максимального элементов её побочной диагонали.

3. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти сумму номеров минимального и максимального элементов её главной диагонали.

4. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти произведение минимального и максимального элементов её главной диагонали. Затем умножить побочную диагональ на максимальный элемент главной диагонали.

5. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти количество положительных элементов её главной диагонали. Затем умножить побочную диагональ на найденное количество.

6. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти среднее арифметическое положительных элементов её побочной диагонали.

7. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти среднее геометрическое положительных элементов её побочной диагонали.

8. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти среднее арифметическое положительных элементов параллели главной диагонали, расположенной выше над диагональю.

9. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти минимальный элемент среди положительных элементов параллели главной диагонали, расположенной выше над диагональю.

10. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти среднее геометрическое отрицательных элементов параллели главной диагонали, расположенной под диагональю.

11. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти произведение отрицательных элементов параллели побочной диагонали, расположенной над диагональю,

12. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти максимальный элемент среди отрицательных элементов параллели побочной диагонали, расположенной над диагональю,

13. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти сумму положительных элементов параллели побочной диагонали, расположенной под диагональю (ниже побочной диагонали).

14. Дан двумерный массив A, размером (n×n)(или квадратная матрица A). Найти сумму и количество положительных элементов параллели побочной диагонали, расположенной под диагональю ( ниже побочной диагонали). Затем каждый элемент побочной диагонали умножить на количество.

15. Дан двумерный массив A, размером (n×n) (или квадратная матрица A). Найти среднее арифметическое положительных элементов параллели главной диагонали, расположенной выше над диагональю.

16. Дан двумерный массив А, размером (n×n) (или квадратная матрица А). Найти среднее геометрическое положительных элементов верхней треугольной матрицы, расположенной выше главной диагонали, исключая саму главную диагональ.