**Algorithmization**

1. **Массив -** порядоченный набор элементов, каждый из которых хранит одно значение. Индексация массива начинается с нуля (a[0]). Обращение к i-тому элементу относится через индекс в квадратных скобках array[i]
2. **Примитивные типы:** int[] array = new int[size] – одномерный массив

int[][] array = new int[size][size] – двумерный массив

**Ссылочные типы – точно также**

**Если не заполнить массив** примитивных типов по умолчанию будет 0 – численные типы, false – логические, ссылочные - null

1. Клонирование массива возвращает ссылку на новый массив, который содержит те же элементы, что и исходный массив.

Если класс имеет только члены примитивных типов, то будет создана совершенно новая копия объекта и возвращена ссылка на этот объект.

Если класс содержит не только члены примитивных типов, а и любого другого типа класса, тогда копируются ссылки на объекты этих классов. Следовательно, оба объекта будут иметь одинаковые ссылки.

1. Двумерный массив представляет собой «массив массивов», образно int[][] array = new int[][]{{1, 2, 3}, {4, 5, 6}}

**Рваный массив (зубчатый)** – массив, у которого различаются длины строк (int [][] array = new array [4][]). Количество строк *array.length,* количество элементов строки через *array[i].length*

1. **java.lang.ArrayIndexOutOfBoundsException –** заданный индекс массива в массиве отсутствует (выход за пределы массива)

**java.lang.ArrayStoreException –** попытка занести в массив значение неправильного типа

1. Для того, что сделать код более читабельным, следовательно увеличить возможность другим программистам работать над ним и расширять проект. Также «отлавливать» ошибки проще через определенные функции.

Для каждой задачи должен быть реализован свой метод с целью избегания проблем «подгонки под ответ», а так модификаций кода

1. Создается копия объекта внутри метода и с ней происходят действия, описанные в теле метода (происходит передача по значению) – **примитивные объекты.**

**Для ссылочных типов** происходит пересылка по ссылке, однако действия метода никак не влияют на исходный объект.

1. Чтобы передать в метод массив, необходимо чтобы массив с нужным типом данных выступал в качестве параметра. Вернуть массив можно с помощью слова return, при этом метод должен иметь тот же тип данных. Изменить размер переданного массива нельзя.
2. Вернуть значение из метода – значит передать значение из тела метода или параметра, передаваемого в метод, с помощью ключевого слова *return.* (тип данных метода и переменной должен совпадать!).

При примитивных типах данных возвращается значение, при ссылочных – ссылка на изначальный объект (т.е. необходимо переопределять)

1. **Сортировка пузырьком (bubble sort):**

If (array[i] > array[i+1]) {

Int temp = array[i];

Array[i] = array[i+1];

Array[i+1] = temp;

}

**Сортировка вставками (insert sort)**

int key;

for (int i = 1; i < array.length; i++) {

key = array[i];

int j = i - 1;

while (j >= 0 && array[j] < key) {

array[j + 1] = array[j];

j = j - 1;

}

array[j + 1] = key;

}

**Сортировка с помощью метода sort класса Arrays**

Arrays.sort(array)

**Сортировка выбором**

for (int i = 0; i < array.length; i++) {

int min = array[i];

int minId = i;

for (int j = i+1; j < array.length; j++) {

if (array[j] < min) {

min = array[j];

minId = j;

}

}

int temp = array[i];

array[i] = min;

array[minId] = temp;

}