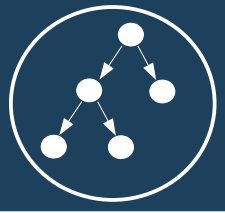
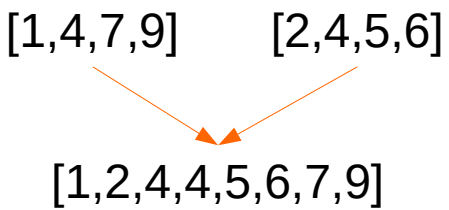


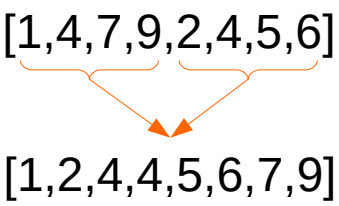
Data Structures and Algorithms

Алгоритмы.
Слияние отсортированных последовательностей

Слияние отсортированных последовательностей

Под слиянием отсортированных последовательностей подразумевают объединение нескольких отсортированных последовательностей в одну отсортированную последовательность.





Слияние может производиться для двух отдельных последовательностей.

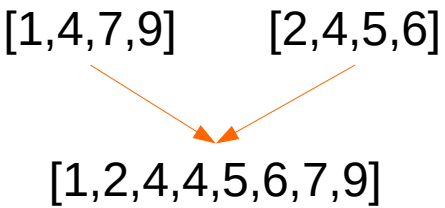
Слияние может производиться для двух отсортированных частей одной последовательностей.

Сведение о алгоритме

Сложность по времени в наихудшем случае $O(n+m)$

Требует дополнительно памяти в размере суммы размеров объединяемых последовательностей.

Слияние отдельных отсортированных последовательностей

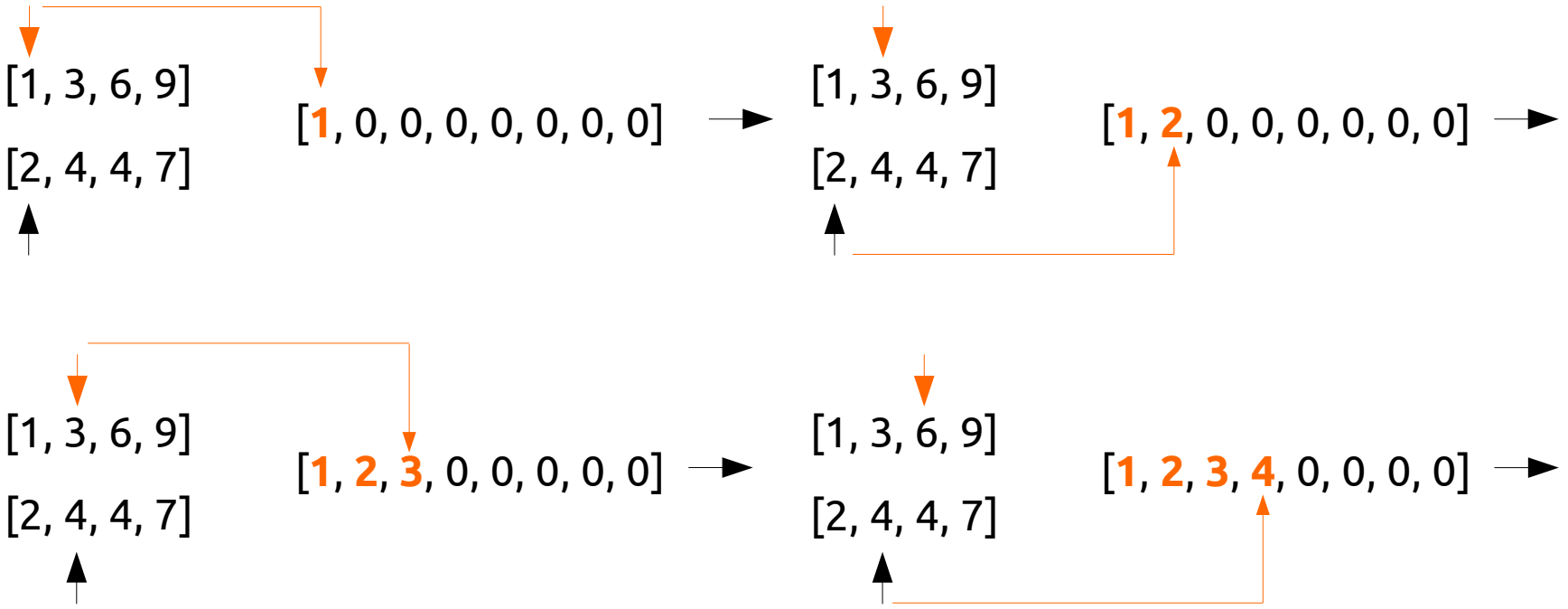


Объединить две отсортированных последовательностей в одну отсортированную последовательность.

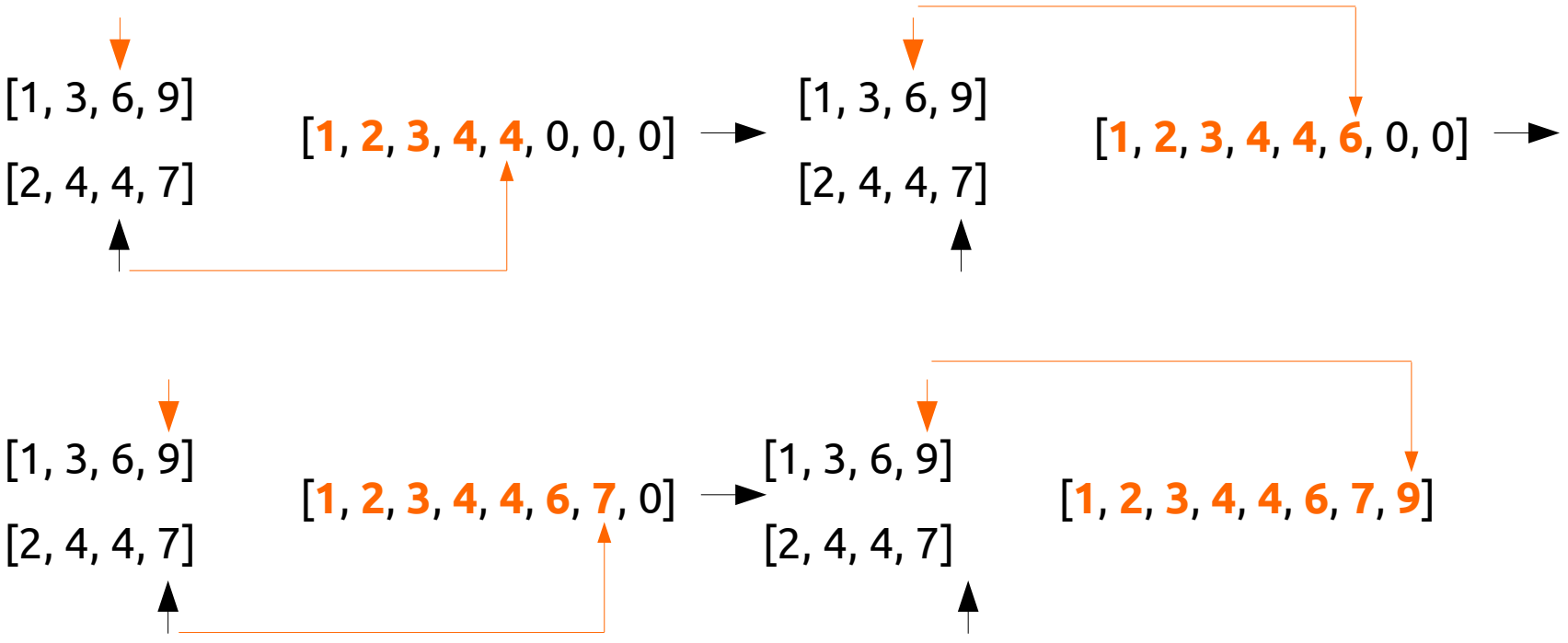
Описание алгоритма

1) Создаем результирующую последовательность длинна которой равна сумме длин объединяемых последовательностей. Создать две вспомогательных переменных для хранения индексов (например l и r) присвоить им значение индексов первого элемента в первой и второй объединяемых последовательностей соответственно.
2) Выполнить проход по результирующей последовательности. Для добавления элемента в результирующую последовательность выполнить ряд проверок:
    1) Если значение индекса l больше длинны первой последовательности то добавить элемент стоящий на индексе r во второй последовательности. Увеличить значение r на единицу.
    2) Если значение индекса r больше длинны второй последовательности то добавить элемент стоящий на индексе l в первой последовательности. Увеличить значение l на единицу.
    3) Если элемент на индексе l меньше элемента на индексе r то добавить элемент стоящий на индексе l в первой последовательности. Увеличить значение l на единицу.
    4) Добавить элемент стоящий на индексе r во второй последовательности. Увеличить значение r на единицу.

Графическое пояснение алгоритма слияния отсортированных последовательностей



Графическое пояснение алгоритма слияния отсортированных последовательностей



Python

Реализация алгоритма на Python

Python

Реализация алгоритма на Python

```
def merge(first_sequince, second_sequince):
    result = [0 for x in range(len(first_sequince)+len(second_sequince))]
    l = 0
    r = 0
    for i in range(len(result)):
        if l >= len(first_sequince):
            result[i] = second_sequince[r]
            r += 1
        elif r >= len(second_sequince):
            result[i] = first_sequince[l]
            l += 1
        elif first_sequince[l] < second_sequince[r]:
            result[i] = first_sequince[l]
            l += 1
        else:
            result[i] = second_sequince[r]
            r += 1
    return result
```

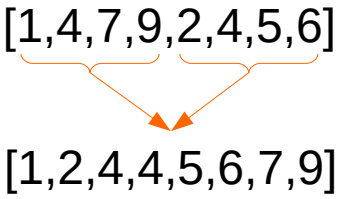
Java

Реализация алгоритма на Java

Реализация алгоритма на Java

```
public static int[] merge(int[] firstArray, int[] secondArray) {
        int[] result = new int[firstArray.length + secondArray.length];
        int l = 0;
        int r = 0;
        for (int i = 0; i < result.length; i++) {
            if (l >= firstArray.length) {
                result[i] = secondArray[r];
                r += 1;
            } else if (r >= secondArray.length) {
                result[i] = firstArray[l];
                l += 1;
            } else if (firstArray[l] < secondArray[r]) {
                result[i] = firstArray[l];
                l += 1;
            } else {
                result[i] = secondArray[r];
                r += 1;
            }
        }
        return result;
    }
```

Слияние отсортированных подпоследовательностей

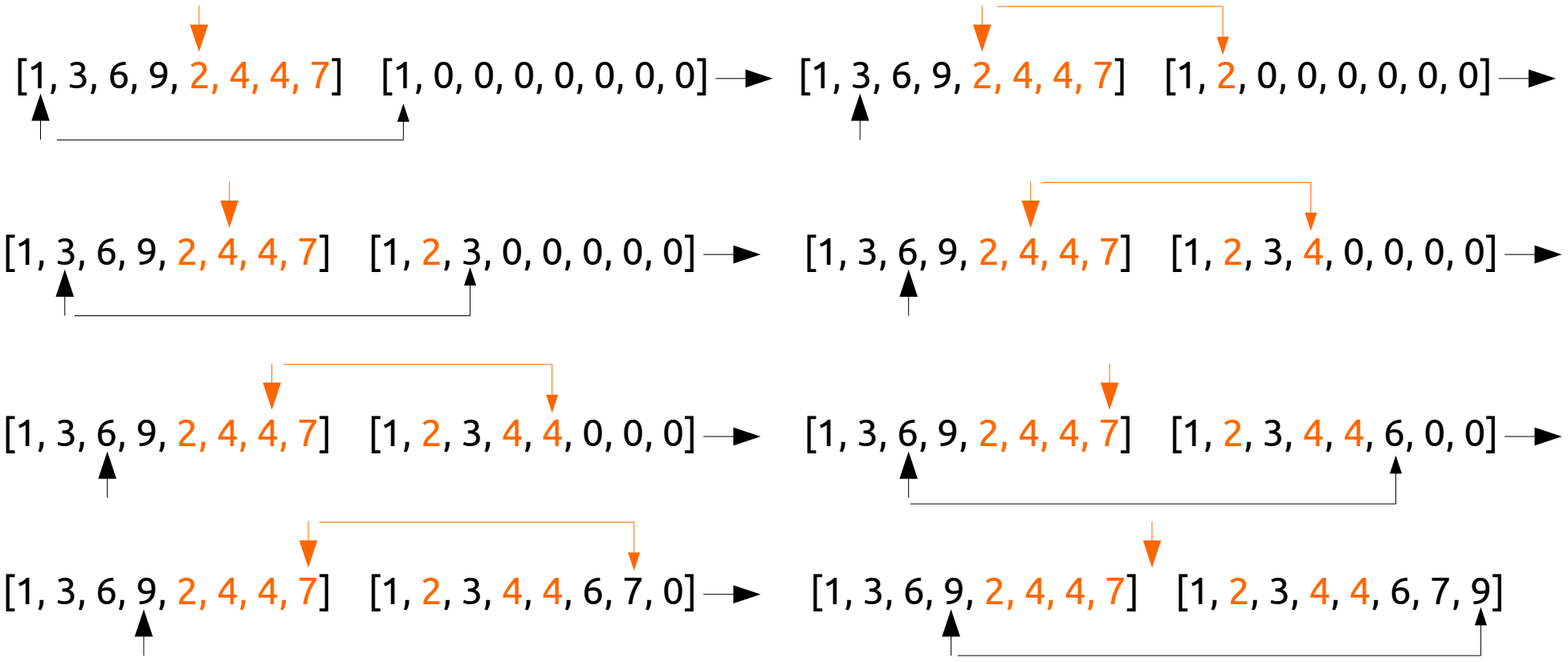


Если в пределах последовательности есть две соседствующие подпоследовательности то их также можно объединить в одну отсортированную последовательность.

Описание алгоритма

1) Создаем результирующую последовательность длинна которой равна длине базовой. Создать две вспомогательных переменных для хранения индексов (например l и r) присвоить им значение индексов первого элемента в первой подпоследовательности и первого элемента во второй подпоследовательности соответственно.
2) Выполнить проход по результирующей последовательности от l до конца второй подпоследовательности. Для добавления элемента в результирующую последовательность выполнить ряд проверок:
    1) Если значение индекса l больше длинны первой подпоследовательности то добавить элемент стоящий на индексе r во второй подпоследовательности. Увеличить значение r на единицу.
    2) Если значение индекса r больше длинны второй подпоследовательности то добавить элемент стоящий на индексе l в первой подпоследовательности. Увеличить значение l на единицу.
    3) Если элемент на индексе l меньше элемента на индексе r то добавить элемент стоящий на индексе l в первой подпоследовательности. Увеличить значение l на единицу.
    4) Добавить элемент стоящий на индексе r во второй подпоследовательности. Увеличить значение r на единицу.

Графическое пояснение алгоритма слияния отсортированных подпоследовательностей



Python

Реализация алгоритма на Python

Python

Реализация алгоритма на Python

```
def merge(sequince, ls, le, rs, re):
    result = sequince[::]
    l = ls
    r = rs
    for i in range(ls, re+1):
        if l > le:
            result[i] = sequince[r]
            r += 1
        elif r > re:
            result[i] = sequince[l]
            l += 1
        elif sequince[l] < sequince[r]:
            result[i] = sequince[l]
            l += 1
        else:
            result[i] = sequince[r]
            r += 1
    return result
```

Java

Реализация алгоритма на Java

Реализация алгоритма на Java

```
public static void merge(int[] array, int ls, int le, int rs, int re) {
        int[] supportArray = Arrays.copyOf(array, array.length);
        int l = ls;
        int r = rs;
        for (int i = ls; i <= re; i++) {
            if (l > le) {
                array[i] = supportArray[r];
                r += 1;
            } else if (r > re) {
                array[i] = supportArray[l];
                l += 1;
            } else if (supportArray[l] < supportArray[r]) {
                array[i] = supportArray[l];
                l += 1;
            } else {
                array[i] = supportArray[r];
                r += 1;
            }
        }
    }
```

Список литературы

1) Роберт Седжвик, Кевин Уэйн «Алгоритмы на java 4-е издание» Пер. с англ. - М. : ООО "И.Д. Вильямс", 2013. ISBN 978-5-8459-1781-2.