

Radix Sort (기수 정렬)

데이터 길이가 같을 때 사용 가능.

[Algorithm](#), [Sort](#)

정수

```
public static int[] radixSort(int[] arr, int maxLen) {
    int bucketNum = 10;

    int divFactor = 1;

    Queue<Integer>[] buckets = new LinkedList[bucketNum];
    for (int j = 0; j < buckets.length; j++)
        buckets[j] = new LinkedList(); // Queue 생성

    for (int i = 0; i < maxLen; i++) {
        for (int digitIndex = 0; digitIndex < arr.length; digitIndex++)
        {
            // N번째 자리 숫자 추출
            int radix = (arr[digitIndex] / divFactor) % 10;

            buckets[radix].offer(arr[digitIndex]);
        }

        int pos = 0;
        for (int bucketIndex = 0; bucketIndex < bucketNum;
bucketIndex++) {
            while (!buckets[bucketIndex].isEmpty()) {
                arr[pos] = buckets[bucketIndex].poll();
                pos++;
            }
        }

        divFactor *= 10;
    }
    return arr;
}
```

문자열

전체 문자열

```

public static String[] radixSort(String[] arr, int maxLen) {
    int bucketNum = Character.MAX_VALUE;
    int charIndex = maxLen - 1; //최대길이 5이면 index 4부터 접근.
    Queue<String>[] buckets = new LinkedList[bucketNum];
    for (int j = 0; j < buckets.length; j++)
        buckets[j] = new LinkedList(); // Queue 생성

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

        for (int digitIndex = 0; digitIndex < arr.length; digitIndex++)
        {
            try {
                String str = arr[digitIndex];

                if(str.length() != maxLen) {
                    throw new Exception "[" + digitIndex + "] 문자열 길이
가 최대 길이와 일치하지 않습니다.");
                }
                int radix = str.charAt(charIndex);
                buckets[radix].offer(arr[digitIndex]);
            } catch (Exception e) {
                System.err.println(e.getMessage());
                return null;
            }
        }

        int pos = 0;
        for (int bucketIndex = 0; bucketIndex < bucketNum;
bucketIndex++) {
            while (!buckets[bucketIndex].isEmpty()) {
                arr[pos] = buckets[bucketIndex].poll();
                pos++;
            }
        }

        charIndex--;
    }
    return arr;
}

```

알파벳

```

public static String[] radixSort(String[] arr, int maxLen) {
    int bucketNum = 'z' - 'a' + 1;
    int charIndex = maxLen - 1; //최대길이 5이면 index 4부터 접근.
    Queue<String>[] buckets = new LinkedList[bucketNum];
    for (int j = 0; j < buckets.length; j++)

```

```

        buckets[j] = new LinkedList(); // Queue 생성

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

            for (int digitIndex = 0; digitIndex < arr.length; digitIndex++)
            {
                try {
                    //소문자로 변환후 비교
                    String str = arr[digitIndex].toLowerCase();

                    if(str.length() != maxLen) {
                        throw new Exception "[" + digitIndex + "] 문자열 길이
가 최대 길이와 일치하지 않습니다.");
                    }
                    int radix = str.charAt(charIndex) - 'a';
                    if(radix < 0 || radix > 'z' - 'a') {
                        throw new Exception "[" + digitIndex + "] 알파벳이 아
닌 문자가 포함되어 있습니다.");
                    }
                    buckets[radix].offer(arr[digitIndex]);
                } catch (Exception e) {
                    System.err.println(e.getMessage());
                    return null;
                }
            }

            int pos = 0;
            for (int bucketIndex = 0; bucketIndex < bucketNum;
bucketIndex++) {
                while (!buckets[bucketIndex].isEmpty()) {
                    arr[pos] = buckets[bucketIndex].poll();
                    pos++;
                }
            }

            charIndex--;
        }
        return arr;
    }

```

From:

<http://ledyx.xyz/> - Ledyx WIKI

Permanent link:

http://ledyx.xyz/doku.php?id=algorithm:radix_sort&rev=1465306107Last update: **2021/02/07 03:15**