http://algs4.cs.princeton.edu

5.1 KEY-INDEXED COUNTING DEMO

Last updated on 11/28/17 7:06 AM

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

R = 6

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

use	a	for	0
	b	for	1
	c	for	2
	d	for	3
	e	for	4
	f	for	5

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

count frequencies

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

offset by 1
[stay tuned]

r	count[r]
a	0
b	2
c	3
d	1
e	2
f	1
–	3

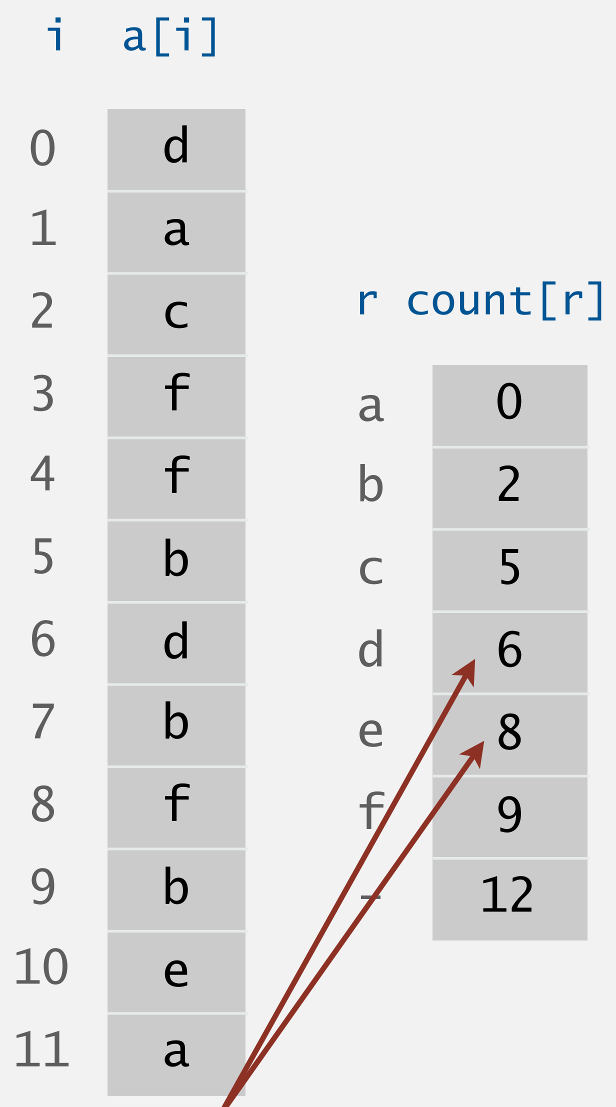
Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

compute cumulates



i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	0
b	2
c	5
d	6
e	8
f	9
-	12

6 keys < d, 8 keys < e
so d's go in a[6] and a[7]

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items $\rightarrow$

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	0
b	2
c	5
d	6
e	8
f	9
–	12

i	aux[i]
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	0
b	2
c	5
d	7
e	8
f	9
-	12

i	aux[i]
0	
1	
2	
3	
4	
5	
6	d
7	
8	
9	
10	
11	

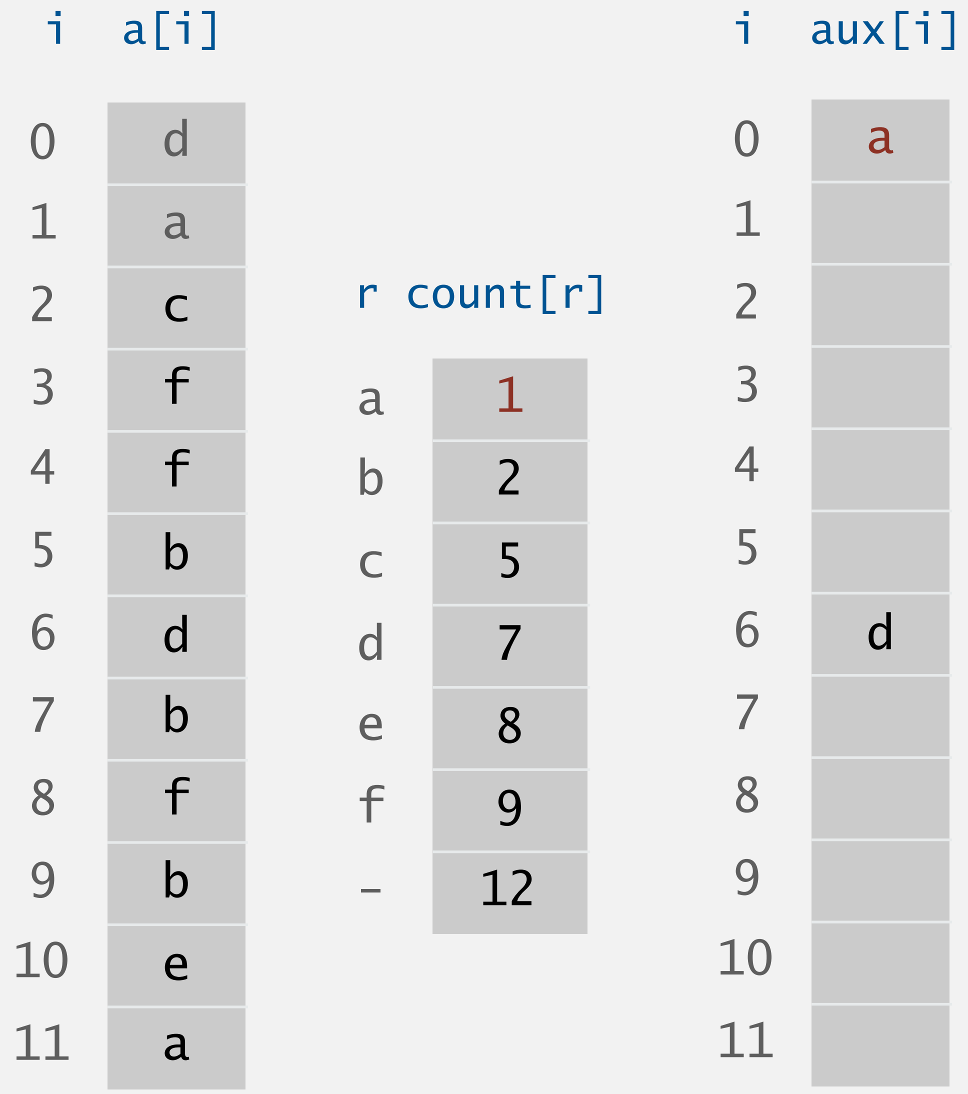
Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items



i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	2
c	5
d	7
e	8
f	9
-	12

i	aux[i]
0	a
1	
2	
3	
4	
5	
6	d
7	
8	
9	
10	
11	

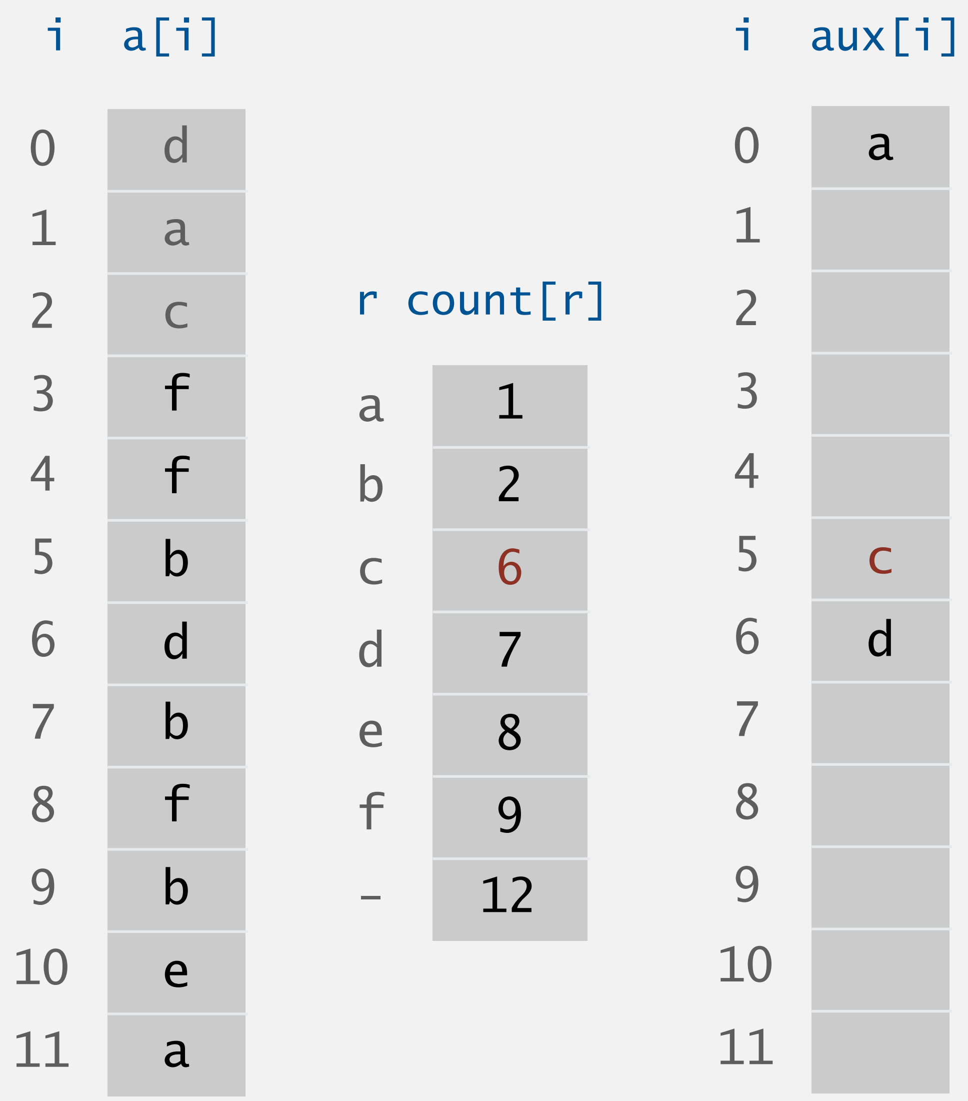
Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

move items

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```



i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	2
c	6
d	7
e	8
f	9
-	12

i	aux[i]
0	a
1	
2	
3	
4	
5	c
6	d
7	
8	
9	
10	
11	

Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items $\longrightarrow$

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	2
c	6
d	7
e	8
f	10
-	12

i	aux[i]
0	a
1	
2	
3	
4	
5	c
6	d
7	
8	
9	f
10	
11	

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	2
c	6
d	7
e	8
f	11
-	12

i	aux[i]
0	a
1	
2	
3	
4	
5	c
6	d
7	
8	
9	f
10	f
11	

Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

move items →

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	3
c	6
d	7
e	8
f	11
-	12

i	aux[i]
0	a
1	
2	b
3	
4	
5	c
6	d
7	
8	
9	f
10	f
11	

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items $\rightarrow$

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	3
c	6
d	8
e	8
f	11
-	12

i	aux[i]
0	a
1	
2	b
3	
4	
5	c
6	d
7	d
8	
9	f
10	f
11	

Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	4
c	6
d	8
e	8
f	11
-	12

i	aux[i]
0	a
1	
2	b
3	b
4	
5	c
6	d
7	d
8	
9	f
10	f
11	

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items $\longrightarrow$

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	4
c	6
d	8
e	8
f	12
-	12

i	aux[i]
0	a
1	
2	b
3	b
4	
5	c
6	d
7	d
8	
9	f
10	f
11	f

Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	5
c	6
d	8
e	8
f	12
-	12

i	aux[i]
0	a
1	
2	b
3	b
4	b
5	c
6	d
7	d
8	
9	f
10	f
11	f

Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

move items

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	1
b	5
c	6
d	8
e	9
f	12
-	12

i	aux[i]
0	a
1	
2	b
3	b
4	b
5	c
6	d
7	d
8	e
9	f
10	f
11	f

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items $\rightarrow$

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	2
b	5
c	6
d	8
e	9
f	12
-	12

i	aux[i]
0	a
1	a
2	b
3	b
4	b
5	c
6	d
7	d
8	e
9	f
10	f
11	f

Key-indexed counting demo

Goal. Sort an array a[] of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

move items →

i	a[i]
0	d
1	a
2	c
3	f
4	f
5	b
6	d
7	b
8	f
9	b
10	e
11	a

r	count[r]
a	2
b	5
c	6
d	8
e	9
f	12
-	12

i	aux[i]
0	a
1	a
2	b
3	b
4	b
5	c
6	d
7	d
8	e
9	f
10	f
11	f

Key-indexed counting demo

Goal. Sort an array `a[]` of n integers between 0 and $R - 1$.

- Count frequencies of each letter using key as index.
- Compute frequency cumulates which specify destinations.
- Access cumulates using key as index to move items.
- Copy back into original array.

```
int n = a.length;
int[] count = new int[R+1];

for (int i = 0; i < n; i++)
   count[a[i]+1]++;

for (int r = 0; r < R; r++)
   count[r+1] += count[r];

for (int i = 0; i < n; i++)
   aux[count[a[i]]++] = a[i];

for (int i = 0; i < n; i++)
   a[i] = aux[i];
```

copy back

i	a[i]
0	a
1	a
2	b
3	b
4	b
5	c
6	d
7	d
8	e
9	f
10	f
11	f

r	count[r]
a	2
b	5
c	6
d	8
e	9
f	12
-	12

i	aux[i]
0	a
1	a
2	b
3	b
4	b
5	c
6	d
7	d
8	e
9	f
10	f
11	f