

归并排序 (merge Sort)

1. 思想: 逐次将已排序的子文件合并, 得到完全排序的文件.

2. 算法: ① 两组归并算法.

将 $r[low]$ 到 $r[m]$ 和 $r[m+1]$ 到 $r[high]$ 两个有序文件合并, 并存储在 $r[low]$ 到 $r[high]$ 中.

内核: void merge (r, r1, low, m, high)

```

{ while (i ≤ m && j ≤ high) {
    if (r[i].key ≤ r[j].key) r[k++] = r[i++];
    else r[k++] = r[j++];
};
while (i ≤ m) r[k++] = r[i++];
while (j ≤ high) r[k++] = r[j++];
}

```

② 一趟归并算法.

内核: void mergePass (r, r1, n, length)

```

{ while (i + 2 * length - 1 < n) {
    merge ( r, r1, i, i + length - 1, i + 2 * length - 1);
    i += 2 * length;
};
if (i + length - 1 < n - 1) (还有两个文件)
    merge ( r, r1, i, i + length - 1, n - 1);
else for (j = i; j < n; j++) r1[j] = r[j]; (直接贴到r1上)
}

```

③ 二路归并算法

内核: void mergeSort (pvector)

```

{ length = 1;
  while (length < pvector.size() -> n) { (最终归并长度不超过n)

```

mergePass (~~record~~ pvector → record, r1, pvector → n, length);

length *= 2;

mergePass (r1, pvector → record, pvector → n, length);

length *= 2;

};

↓

3. 图示

(n=9)

length

①	<u>25</u>	<u>57</u>	<u>48</u>	<u>37</u>	<u>12</u>	<u>82</u>	<u>75</u>	<u>29</u>	<u>16</u>	
②	25	57	37	48	12	82	29	75	16	1
③	25	37	48	57	12	29	75	82	16	2
④	12	25	29	37	48	57	75	82	16	4
⑤	12	16	25	29	37	48	57	75	82	8

空间 $O(n)$

时 $O(n \log_2 n)$

稳定