/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

直接上模板，不多BB

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int temp[100005]; // 存中间数据

void insort(int v[], int l, int r)

{

int tmp;

for (int i = l + 1, j; i <= r; i++)

{

tmp = v[i]; j = i - 1;

while (j >= l && v[j] > tmp)

v[j + 1] = v[j], j--;

v[j + 1] = tmp;

}

}

void merge(int v[], int l, int r)

{

int m = (l + r) >> 1;

int k = l, i = l, j = m + 1;

while (i <= m && j <= r)

{

if (v[i] < v[j])

temp[k] = v[i], k++, i++;

else

temp[k] = v[j], k++, j++;

}

while (i <= m)

temp[k] = v[i], k++, i++;

while (j <= r)

temp[k] = v[j], k++, j++;

for (i = l; i <= r; i++)

v[i] = temp[i];

}

void msort(int v[], int l, int r)

{

if (l + 2 > r)

insort(v, l, r);

else

{

int m = (l + r) >> 1;

msort(v, l, m);

msort(v, m + 1, r);

if (v[m] > v[m + 1])

merge(v, l, r);

}

}