### 归并排序

#include <cstdio>  
#include <cstring>  
  
template<typename T>  
void mergeSort(T arr[], T reg[], int start, int end) {  
 if (start >= end)  
 return;  
 int len = end - start, mid = (len >> 1) + start;  
 int start1 = start, end1 = mid;  
 int start2 = mid + 1, end2 = end;  
 mergeSort(arr, reg, start1, end1);  
 mergeSort(arr, reg, start2, end2);  
 int k = start;  
 while (start1 <= end1 && start2 <= end2)  
 reg[k++] = arr[start1] <= arr[start2] ? arr[start1++] : arr[start2++];  
 while (start1 <= end1)  
 reg[k++] = arr[start1++];  
 while (start2 <= end2)  
 reg[k++] = arr[start2++];  
 for (k = start; k <= end; k++)  
 arr[k] = reg[k];  
}  
  
long long res = 0;  
template<typename T>  
void reverse\_arr(T arr[], T reg[], int start, int end) {  
 if (start >= end)  
 return;  
 int len = end - start, mid = (len >> 1) + start;  
 int start1 = start, end1 = mid;  
 int start2 = mid + 1, end2 = end;  
 reverse\_arr(arr, reg, start1, end1);  
 reverse\_arr(arr, reg, start2, end2);  
 int k = start;  
 while (start1 <= end1 && start2 <= end2) {  
 if (arr[start1] <= arr[start2])  
 reg[k++] = arr[start1++];  
 else {  
 reg[k++] = arr[start2++];  
 res += (mid - start1 + 1);  
 }  
 }  
 while (start1 <= end1)  
 reg[k++] = arr[start1++];  
 while (start2 <= end2)  
 reg[k++] = arr[start2++];  
 for (k = start; k <= end; k++)  
 arr[k] = reg[k];  
}  
  
int main()  
{  
 int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 0};  
 int tem[10];  
 memset(tem, 0, sizeof(tem));  
 reverse\_arr(arr, tem, 0, 9);  
 return 0;  
}