/\*\*

\* MergeSort.c 并归排序算法之递归实现

\*

\* @author Darbuly 970073804@qq.com

\* @copyright 2019-2020

\*/

#include <stdio.h>

#define MAXSIZE 10

void merging(int \*list1,int list1\_size,int \*list2,int list2\_size)

{

int i,j,k,m;

int temp[MAXSIZE];

i = j = k = 0;

while(i<list1\_size &&j<list2\_size)

{

if(list1[i]<list2[j])

{

temp[k++] = list1[i++];

}

else

{

temp[k++] = list2[j++];

}

}

while(i<list1\_size)

{

temp[k++] = list1[i++];

}

while(j<list2\_size)

{

temp[k++] = list2[j++];

}

for(m=0;m<(list1\_size + list2\_size);m++)

{

list1[m] = temp[m];

}

}

void MergeSort(int k[],int n)

{

if(n>1)

{

int \*list1 = k;

int list1\_size = n/2;

int \*list2 = k + n/2;

int list2\_size = n - list1\_size;

MergeSort(list1,list1\_size);

MergeSort(list2,list2\_size);

merging(list1,list1\_size,list2,list2\_size);

}

}

int main()

{

int i,a[10] = {5,2,6,0,3,9,1,7,4,8};

printf("MergeSort.c Test\n");

MergeSort(a,10);

printf("Sorted BY MergeSort:\n");

for(i=0;i<10;i++)

{

printf("%d",a[i]);

}

printf("\n\n");

return 0;

}

/\*\*

\* MergeSort2.c 并归排序算法 迭代实现

\*

\* @author Darbuly 2019-2020

\*

\*/

#include <stdio.h>

#include <stdlib.h>

#define MAXSIZE 10

void MergeSort(int k[],int n)

{

int i,left\_min,left\_max,right\_min,right\_max,next;

int \*temp = (int \*)malloc(n\*sizeof(int));

for(i=1;i<n;i\*=2)

{

for(left\_min=0;left\_min<n-i;left\_min = right\_max)

{

right\_min =left\_max = left\_min + i;

right\_max = right\_min + i;

if(right\_max > n)

{

right\_max = n;

}

next = 0;

while(left\_min<left\_max&&right\_min<right\_max)

{

if(k[left\_min]<k[right\_min])

{

temp[next++] = k[left\_min++];

}

else

{

temp[next++] = k[right\_min++];

}

}

while(left\_min<left\_max)

{

k[--right\_min] = k[--left\_max];

}

while(next>0)

{

k[--right\_min] = temp[--next];

}

}

}

}

int main()

{

int i,a[10] = {5,2,6,0,3,9,1,7,4,8};

printf("MergeSort.c Test\n");

MergeSort(a,10);

printf("Sorted BY MergeSort:\n");

for(i=0;i<10;i++)

{

printf("%d",a[i]);

}

printf("\n\n");

return 0;

}