#include "sorting.h"

using namespace std;

namespace csci\_591\_sorting

{

void Sorting::swap(int& a, int& b)

{

int temp = a;

a = b;

b = temp;

}

void Sorting::QuickSort ( Item c[], int first, int last )

{

int pivotElement;

if(first < last)

{

pivotElement = pivot(c, first, last);

QuickSort(c, first, pivotElement-1);

QuickSort(c, pivotElement+1, last);

}

}

int Sorting::pivot(int a[], int first, int last)

{

int p = first;

int pivotElement = a[first];

for(int i = first+1 ; i <= last ; i++)

{

used++;

if(a[i] <= pivotElement)

{

p++;

swap(a[i], a[p]);

}

}

swap(a[p], a[first]);

return p;

}

void Sorting::InsertionSort(int arr[], size\_t high)

{

for (int i=1; i<=high; i++)

{

int key = arr[i];

int j = i;

used++;

while (j > 0 && arr[j-1] > key)

{

arr[j] =arr[j-1];

j--;

used++;

}

arr[j] = key;

} }

void Sorting::MergeSort(Item Array[], size\_t Left, size\_t Right)

{

if(Left < Right)

{

int MidVal= (Left+Right)/2;

MergeSort(Array, Left, MidVal);

MergeSort(Array, MidVal+1, Right);

Merge(Array, Left, MidVal, Right);

}

}

void Sorting::Merge(Item Array[], size\_t Left, size\_t Mid, size\_t Right)

{

int i, j, k;

int SizeLeft = Mid - Left + 1;

int SizeRight = Right - Mid;

int Lft[SizeLeft];

int Rght[SizeRight];

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

{

used++;

Lft[i] = Array[Left+i];

}

for(j = 0; j < SizeRight; j++)

{

used++;

Rght[j] = Array[Mid+1+j];

}

i = 0;

j = 0;

k = Left;

while (i < SizeLeft && j < SizeRight)

{

used++;

if (Lft[i] <= Rght[j])

{

Array[k] = Lft[i];

i++;

}

else

{

Array[k] = Rght[j];

j++;

}

k++;

}

while (i < SizeLeft)

{

Array[k] = Lft[i];

i++;

k++;

used++;

}

while (j < SizeRight)

{

Array[k] = Rght[j];

j++;

k++;

used++;

}

}

}