int k;

for (int i = 0; i < n-1; i++) {

k = i;

for (int j = i+1; j < n; j++)

if (a[j] < a[k])

k = j;

hoandoi(a[k], a[i]);

}

Sap xep theo SelectionSort:

9 3 5 1 6 8 2 4 7

-> Buoc 1: 1| 3 5 9 6 8 2 4 7

-> Buoc 2: 1 2| 5 9 6 8 3 4 7

-> Buoc 3: 1 2 3| 9 6 8 5 4 7

-> Buoc 4: 1 2 3 4| 6 8 5 9 7

-> Buoc 5: 1 2 3 4 5| 8 6 9 7

-> Buoc 6: 1 2 3 4 5 6 |8 9 7

-> Buoc 7: 1 2 3 4 5 6 7| 9 8

-> Buoc 8: 1 2 3 4 5 6 7 8 9|

int pos, i;int x;

for (int i = 1; i < n; i++) {

x = a[i]; pos = i - 1;

**while** ((pos >= 0) && (a[pos] > x)){

a[pos+1] = a[pos];

pos--;

}

a[pos+1] = x;

}

Sap xep theo InsertionSort:

9 3 5 1 6 8 2 4 7

-> Buoc 2: 3 9 5 1 6 8 2 4 7

-> Buoc 3: 3 5 9 1 6 8 2 4 7

-> Buoc 4: 1 3 5 9 6 8 2 4 7

-> Buoc 5: 1 3 5 6 9 8 2 4 7

-> Buoc 6: 1 3 5 6 8 9 2 4 7

-> Buoc 7: 1 2 3 5 6 8 9 4 7

-> Buoc 8: 1 2 3 4 5 6 8 9 7

-> Buoc 9: 1 2 3 4 5 6 7 8 9

int i,j;

    for(i = 0; i < n-1; i++){

        for(j = i+1; j < n; j++)

            if(a[i] > a[j]) *//nếu sai vị trí thì đổi chỗ*

                hoandoi(a[i], a[j]);

}

Sap xep theo InterchangeSort:

9 3 5 1 6 8 2 4 7

-> Buoc 1: 1 9 5 3 6 8 2 4 7

-> Buoc 2: 1 2 9 5 6 8 3 4 7

-> Buoc 3: 1 2 3 9 6 8 5 4 7

-> Buoc 4: 1 2 3 4 9 8 6 5 7

-> Buoc 5: 1 2 3 4 5 9 8 6 7

-> Buoc 6: 1 2 3 4 5 6 9 8 7

-> Buoc 7: 1 2 3 4 5 6 7 9 8

-> Buoc 8: 1 2 3 4 5 6 7 8 9

int i,j;

for(i=0; i<(n-1);i++){

for(j=n-1; j> i; j--){

if(a[j] < a[j-1])

hoandoi(a[j], a[j-1]);

Sap xep theo BubbleSort:

> Buoc 1: 1 9 3 5 2 6 8 4 7

> Buoc 2: 1 2 9 3 5 4 6 8 7

> Buoc 3: 1 2 3 9 4 5 6 7 8

> Buoc 4: 1 2 3 4 9 5 6 7 8

> Buoc 5: 1 2 3 4 5 9 6 7 8

> Buoc 6: 1 2 3 4 5 6 9 7 8

> Buoc 7: 1 2 3 4 5 6 7 9 8

> Buoc 8: 1 2 3 4 5 6 7 8 9

#include <stdio.h>

#include <math.h>

#include <assert.h>

#include <time.h>

void Nhapmang(int a[], int n);

void Xuatmang(int a[], int n);

void hoandoi(int &x, int &y);

void SelectionSort (int a[], int n);

void InsertionSort (int a[], int n);

void InterchangeSort( int a[], int n);

void BubbleSort(int a[], int n);

int partition (int a[], int l, int r);

void quicksort (int a[], int l, int r);

void merge\_sort\_recursion(int a[], int l, int r);

void merge\_sort(int a[], int l, int m, int r);

void heapify (int a[], int n, int i);

void heapsort (int a[], int n);

void shellsort(int a[], int n);

int randint (int max);

void createRamdomArr(int a[], int n, int maximum);

int main ()

{

//int n; int a[100];

int n = 9; int a[100] = {9, 3, 5, 1, 6, 8, 2, 4, 7};

// printf("nhap vao so phan tu cua mang: ");scanf("%d", &n);Nhapmang(a, n);

printf("Mang ban dau chua sap xep: "); Xuatmang(a, n);

int opt; clock\_t start = clock();

printf("\nChon phuong phap Sap xep: ");scanf("%d", &opt);

switch (opt)

{

case 1: SelectionSort(a, n);

break;

case 2: InsertionSort(a, n);

break;

case 3: InterchangeSort(a, n);

break;

case 4: BubbleSort(a, n);

break;

case 5: quicksort(a, 0, n - 1);

break;

case 6: merge\_sort\_recursion(a, 0, n - 1);

break;

case 7: heapsort(a, n);

break;

case 8: shellsort(a, n);

break;

}

printf("\nMang sau khi da sap xep: "); Xuatmang(a, n);

createRamdomArr(a, n, 1000);

clock\_t end = clock();

double t = (double)(end - start) / CLOCKS\_PER\_SEC;

printf("Thoi gian thuc hien : %.3f", t);

return 0;

}

void Nhapmang(int a[], int n)

{

for (int i = 0; i < n ; i++)

{

printf("Nhap phan tu thu %d: ", i);

scanf("%d", &a[i]);

}

}

void Xuatmang(int a[], int n)

{

for (int i = 0; i < n; i++)

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

printf("\n");

}

void hoandoi(int &x, int &y)

{

int z = x;

x = y;

y = z;

}

void SelectionSort (int a[], int n)

{

printf("\n Sap xep theo SelectionSort: \n");

int k;

for(int i = 0; i < n-1; i++)

{

k = i;

for(int j = i+1; j < n; j++)

if (a[j] < a[k])

k = j;

hoandoi(a[k], a[i]);

printf("-> Buoc %d: ", i+1);

for (int t = 0; t < n; t++)

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

printf("\n");

}

}

void InsertionSort (int a[], int n)

{

printf("\n Sap xep theo InsertionSort: \n");

int pos, i;

int x;

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

{

x = a[i]; pos = i - 1;

while ((pos >= 0) && (a[pos] > x))

{

a[pos+1] = a[pos];

pos--;

}

a[pos+1] = x;

printf("-> Buoc %d: ", i+1);

for (int t = 0; t < n; t++)

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

printf("\n");

}

}

void InterchangeSort( int a[], int n)

{

printf("\n Sap xep theo InterchangeSort: \n");

int i,j;

for(i = 0; i < n-1; i++){

for(j = i+1; j < n; j++){

if(a[i] > a[j]) //nếu sai vị trí thì đổi chỗ

hoandoi(a[i], a[j]);

printf("-> Buoc %d: ", i+1);

for (int t = 0; t < n; t++)

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

printf("\n");

}

printf("> Buoc cuoi %d: ", i+1);

for (int t = 0; t < n; t++)

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

printf("\n\n");

}

}

void BubbleSort(int a[], int n)

{

printf("\n Sap xep theo BubbleSort: \n");

int i,j;

for(i=0; i<(n-1);i++){

for(j=n-1; j> i; j--){

if(a[j] < a[j-1])

hoandoi(a[j], a[j-1]);

printf("-> Buoc %d: ", i+1);

for (int t = 0; t < n; t++)

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

printf("\n");

}

printf("> Buoc cuoi %d: ", i+1);

for (int t = 0; t < n; t++)

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

printf("\n\n");

}

}

int partition (int a[], int l, int r)

{

int pivot = a[r];// vi tri cuoi cung ben phai

int i = l - 1;

for (int j = l; j < r; j++)

{

if (a[j] <= pivot)

{

++i;

hoandoi(a[i], a[j]);

}

printf("-> Buoc %d: ", i+1);

for (int t = 0; t < r+1; t++)

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

printf("\n");

}// dua chot ve giua =))

++i;

hoandoi(a[i], a[r]);

return i; // vi tri

}

void quicksort (int a[], int l, int r)

{

if (l >= r) return;

int p = partition(a, l , r);

quicksort(a, l, p - 1);

quicksort(a, p + 1, r);

}

void merge\_sort\_recursion(int a[], int l, int r)

{

if (l < r)

{

int m = l + (r - l) / 2;

merge\_sort\_recursion(a, l, m);

merge\_sort\_recursion(a, m + 1, r);

merge\_sort(a, l , m, r);

}

}

void merge\_sort(int a[], int l, int m, int r)

{

int n\_left = m - l + 1;

int n\_right = r - m;

int left\_arr[n\_left];

int right\_arr[n\_right];

int i, j, k;

for (i = 0; i < n\_left; i++)

left\_arr[i] = a[l+i];

for (i = 0; i < n\_right; i++)

right\_arr[i] = a[m + 1 + i];

for (i = 0, j = 0, k = l; k <= r; k++)

{

if ((i < n\_left) &&

(j >= n\_right || (left\_arr[i] <= right\_arr[j])))

{

a[k] = left\_arr[i];

i++;

} else

{

a[k] = right\_arr[j];

j++;

}

}

}

void heapify (int a[], int n, int i)

{

int largest = i;

int l = 2 \* i + 1;

int r = 2 \* i + 2;

if (l < n && a[l] > a[largest])

largest = l;

if (r < n && a[r] > a[largest])

largest = r;

if (largest != i)

{

hoandoi(a[i], a[largest]);

heapify(a, n, largest);

}

}

void heapsort (int a[], int n)

{

for (int i = n / 2 -1; i >= 0; i--)

heapify(a, n, i);

for (int i = n -1; i >= 0; i--){

hoandoi(a[0], a[i]);

heapify(a, i, 0);

}

}

void shellsort(int a[], int n)

{

int h = n /2;

while (h > 0)

{

for (int i = h; i < n; i++)

{

int x = a[i];

int j = i;

while (a[j - h] > x && j > h - 1)

{

a[j] = a[j - h];

j -= h;

}

a[j] = x;

}

h /= 2;

}

}

int randint (int max)

{

if (max -1 == RAND\_MAX) return rand();

else

{

long end = RAND\_MAX / max;

assert (end > 0L);

end \*= max;

int r;

while ((r = rand()) >= end);

return r % max;

}

}

void createRamdomArr(int a[], int n, int maximum)

{

for (int i = 0; i < n; i++)

a[i] = randint(maximum);

}