#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define MAX 15

#define N 6 //rows

#define M 4 // columns

void init(int mas[][M], int n);

int \* initD(int \* mas, int n, int m);

void print(int mas[][M], int n);

void printD(int \* mas, int n, int m);

int randnum(int min, int max);

void swap(int \* x, int \* y);

void SelectSort(int ar[], int n);

void SelectSortAll(int mas[][M], int n);

int \* SWAPMatrixOnD(int mas[][M], int n);

void mergeSortD(int \* mas, int n, int m);

void printDstr(int \* mas, int n, int m);

int main(void)

{

int tmas[N][M];

init(tmas, N);

print(tmas, N);

SelectSortAll(tmas, N);

print(tmas, N);

int \* mat = SWAPMatrixOnD(tmas, N);

printD(mat, N, M);

mergeSortD(mat, N, M);

printDstr(mat, N, M);

getchar();

return 0;

}

int \* initD(int \* mas, int n, int m)

{

srand(time(NULL));

mas = (int \*)malloc(sizeof(int) \* (m \* n));

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

{

for (int j = 0; j < m; ++j)

{

\*(mas + i \* m + j) = randnum(1, 100);

}

}

return mas;

}

void mergeSortD(int \* mas, int n, int m)

{

int step = m;

int \*ar = (int\*)malloc((n \* m) \* sizeof(int));

while (step < (n \* m))

{

int index = 0;

int l = 0;

int mid = l + step;

int r = l + step \* 2;

do

{

if (mid >= (n \* m))

mid = (n \* m);

if (r >= (n \* m))

r = (n \* m);

int i1 = l, i2 = mid;

for (; i1 < mid && i2 < r; )

{

if (mas[i1] < mas[i2])

ar[index++] = mas[i1++];

else

ar[index++] = mas[i2++];

}

while (i1 < mid)

ar[index++] = mas[i1++];

while (i2 < r)

ar[index++] = mas[i2++];

l += step \* 2;

mid += step \* 2;

r += step \* 2;

} while (l < (n \* m));

for (int i = 0; i < (n \* m); i++)

mas[i] = ar[i];

step \*= 2;

}

free(ar);

}

int \* SWAPMatrixOnD(int mas[][M], int n)

{

int \* ar = (int\*)malloc(sizeof(int) \* (n \* M));

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

{

for (int j = 0; j < M; ++j)

{

\*(ar + i \* M + j) = mas[i][j];

}

}

return ar;

}

void init(int mas[][M], int n)

{

srand(time(NULL));

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

{

for (int j = 0; j < M; ++j)

{

mas[i][j] = randnum(1, 100);

}

}

}

void print(int mas[][M], int n)

{

putchar('\n');

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

{

for (int j = 0; j < M; ++j)

{

printf(" %3d", mas[i][j]);

}

printf("\n\n");

}

}

int randnum(int min, int max)

{

return min + rand() % (max - min + 1);

}

void swap(int \*x, int \*y)

{

int temp = \*x;

\*x = \*y;

\*y = temp;

}

void SelectSort(int mass[], int size)

{

int i, j;

for (i = 0; i < size - 1; ++i)

{

for (j = i + 1; j < size; ++j)

{

if (mass[i] > mass[j])

{

swap(mass + i, mass + j);

}

}

}

}

void SelectSortAll(int mas[][M], int n)

{

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

{

SelectSort(\*(mas + i), M);

}

}

void printD(int \* mas, int n, int m)

{

putchar('\n');

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

{

for (int j = 0; j < m; ++j)

{

printf(" %3d", \*(mas + i \* m + j));

}

printf("\n\n");

}

}

void printDstr(int \* mas, int n, int m)

{

{

putchar('\n');

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

{

for (int j = 0; j < m; ++j)

{

printf(" %3d", \*(mas + i \* m + j));

}

}

}

}