#include<iostream>

using namespace std;

//int\*\* reserveArray(int row,int col) {

// auto arr = new int\* [row] {};

// for (size\_t i = 0; i < row; i++)

// {

// arr[i] = new int[col] {};

// }

// return arr;

//}

//

//void initArray(int\*\* arr, int &row, int &col) {

// for (size\_t i = 0; i < row; i++)

// {

// for (size\_t k = 0; k < col; k++)

// {

// arr[i][k] = (i + 1) \* 10 + k + 1;

// }

// }

//}

//

//

//void printArray(int\*\* arr, int& row, int& col) {

// for (size\_t i = 0; i < row; i++)

// {

// for (size\_t k = 0; k < col; k++)

// {

// cout << arr[i][k] << " ";

// }cout << endl;

// }

// cout << endl<<endl;

//}

//

//int\*\* getOptimiseArray(int\*\* arr, int& row, int& col) {

// auto newarray = reserveArray(row / 2,col);

// for (size\_t i = 0; i < row/2; i++)

// {

// for (size\_t k = 0; k < col; k++)

// {

// newarray[i][k] = arr[i][k] + arr[row - i - 1][k];

// }

// }

// row = row / 2;

// return newarray;

//}

//

//void main() {

//

// int row = 0;

// cin >> row;

// int col = 0;

// cin >> col;

// auto arr = reserveArray(row,col);

// initArray(arr, row, col);

// printArray(arr, row, col);

//

// while (row!=1)

// {

// arr = getOptimiseArray(arr, row, col);

// printArray(arr, row, col);

//

// }

//

//}

//

//Bit Fields

//struct Validation {

// bool isEmailValidate : 1;

// bool isPhoneValidate : 1;

// bool isAgeValidate : 1;

// bool isCardValidate : 1;

// bool isEmailValidate2 : 1;

// bool isPhoneValidate2 : 1;

// bool isAgeValidate2 : 1;

// bool isCardValidate2 : 1;

//};

//

//void main() {

// Validation v;

// v.isEmailValidate = false;

// cout << v.isEmailValidate << endl;

// cout << sizeof(Validation) << endl;

//}

//struct User {

// short age:10;

// short score:7;

//};

//

//void main() {

// User user{ 150,44 };

// cout << user.age << endl;

// cout << user.score << endl;

// cout << sizeof(User) << endl;

//}

//Standart struct

//Nested struct or Embedded

//struct Date {

// short day:6;

// short month:5;

// short year;

//};

// // 16 16 =>32=>4

// //2 2 2 =>

//struct Time {

// short second:7;

// short minute : 7;

// short hour : 6;

//};

//

//struct Event {

// char\* name;

// char\* location;

// Date date;

// Time time;

//};

struct HDD {

int filesCount = 0;

double maxSize = 1024;//byte

double size = 0;

struct File {

char\* name;

char\* content;

};

File\*\* files;

};

//File\*\* files = new File \* [filesCount + 1];

//nezere alin file yaranmamishdan once hemen adda fayl varsa yaranmasin

//eger fayl disce yerleshmirse yaranmasin

//hech bir problem yoxdursa Files massivine hemen fayl elave olunsun

//ve discin tutumu deyishsin

void createFile(HDD& disc, const char\* filename, const char\* content) {

//HDD::File myfile{ filename,content };

//AddFileToDisc(disc, myfile);

}

//nezere alin file update once hemen adda fayl yoxdusa

//eger fayl disce yerleshmirse yaranmasin

//hech bir problem yoxdursa Files massivine hemen fayl elave olunsun

//ve discin tutumu deyishsin

void updateFile(HDD& disc, const char\* filename, const char\* newcontent) {

}

void main() {

// :: scope resolution

/\*HDD::File hdd;\*/

HDD mydisc;

//createFile(mydisc, "a.txt", "Salam Millet");

//createFile(mydisc, "a.txt", "Salam Millet");

}