Task 1. Heavy coins

In the lowland kingdom there was a magnificent wall that for centuries protected its territory from invasions. However, exactly one year ago a gang of robbers decided that the wall was too complicated and burned it to the ground. Now princess Maria has planned to hire skilled builders to restore the wall to its original glory. To pay the builders, she went to the royal treasury and found $N = 2^K$ coins, each of which has a unique number from 0 to $N - 1$. The court accountant informed her that there were two types of coins - light and heavy, with all heavy coins being equally heavy as well as all light coins. Also, according to the accountant, there is at least one heavy and at least one light coin in the treasury.

According to the terms agreed upon with the builders of the new wall, princess Maria has to pay them only in heavy coins, and before the work can begin, she owes an advance of one heavy coin. The problem is that the accountant does not know how many and which of the coins are heavy. Fortunately, he has scales with which he can compare the combined weight of two sets consisting of the same number of coins. The accountant now has to make sure that the number of heavy coins in the royal treasury would be sufficient for the payment and has to find one such coin with which the advance can be paid. However, this would take him too much time, and princess Maria wants the construction of the new wall to begin as soon as possible, so she is turning to you for help.

Task

Write a program `coins`, implementing the function *`count_heavy`*, which will be compiled with the jury's program and through communication with it, finds the total number of the heavy coins in the royal treasury as well as the number of one such coin.

Implementation details

Your function should have the following format:

```
pair<int, int> count_heavy(int k)
```

It will receive as a parameter the number K (we remind you that the number of coins is 2^K) and have to return a pair of two integers – the number of heavy coins and the unique number of one such coin.

To communicate with the jury program, the following function is provided to you:

```
int weigh(const vector<int> &a, const vector<int> &b)
```

It takes as parameters two vectors containing the numbers of the coins for the weighing. The vectors must have an equal number of elements and must not contain repeating numbers. The function will return a value:

- 0, if the two sets of coins have equal total weight;
- -1, if the total weight of the coins whose numbers are in vector a is less than the total weight of the coins whose numbers are in b;
- $+1$, if the total weight of the coins whose numbers are in vector a is more than the total weight of the coins whose numbers are in b.

Note that the complexity of the function *`weigh`* is proportional to the total number of elements in the vectors received as parameters.

You must submit the file `coins.cpp` to the system which contains the *`count_heavy`* function. It may contain other code and functions necessary for your work, but it must not contain the *`main`* function. Also, you must not read from the standard input or print to the standard output. Your program must also include the `coins.h` header file by instruction to the preprocessor:

```
#include "coins.h"
```

Constraints

$1 \leq K \leq 20$

Scoring

If, during its execution, your solution gives a correct answer to a test using no more than $10\,000$ calls to the *weigh* function, a coefficient will be calculated for that test according to the formula $coef = min\left(1, \frac{11}{9} * \left(\frac{author+1}{yours+1}\right)\right)$, where $author$ is the **maximum** count of calls to the *weigh* function, performed by one of the intended author's solutions (see subtasks below), while $yours$ is the count of the call to the *weigh* function, performed by your solution. Otherwise, for that test $coef$ will have a value of 0.

Each subtask consists of a certain number of groups, each group having exactly three tests. Each of the groups is evaluated **independently**, and the number of points, you will receive for it, is equal to the product of the number of points provided for it and the minimum value of the coefficient $coef$ for a test in this group.

Subtasks

Subtask	Points	$author$	Other constraints
1	5	≤ 40	$K \leq 5$
2	10	≤ 250	There is only one heavy or one light coin.
3	20	≤ 1000	There are no more than K heavy or no more than K light coins.
4	25	≤ 100	$K \leq 10$
5	40	≤ 500	No further constraints.

Sample communication

Contestant function	The jury's program
	Calls count_heavy(2)
Calls weigh({0, 1}, {2, 3})	Returns value -1
Calls weigh({1, 2}, {0, 3})	Returns value +1
Calls weigh({1}, {2})	Returns value 0
Returns {3, 2}	

Local testing

For local testing you are provided with the files `coins.h` and `Lgrader.cpp`. Place your file `coins.cpp` and the two provided files in the same folder. Then compile the three files together. In such a way, you will obtain a program to check the correctness of your function. The program will require the following sequence of data:

- on the first line: one positive integer denoting the value of K.
- on the second line: 2^K numbers with values 1 or 0 depending if the corresponding coin is heavy or light.

The program will output information about the calls of the *weigh* function, the answer that your function *count_heavy* has returned and a message indicating whether the answer is correct or not.