/\*\*

\*

\* Solution to homework assignment 1

\* Introduction to programming course

\* Faculty of Mathematics and Informatics of Sofia University

\* Winter semester 2023/2024

\*

\* @author Petya Petrova

\* @idnumber 1MI0600311

\* @task 1

\*

\*/

#include <iostream>

using namespace std;

bool validationOfNumbers(int n, int k) {

if (n < k) {

return 0;

}

return 1;

}

unsigned int divisorsOfNumber(int number) {

if (number == 1) {

return 1;

}

int count = 0;

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

if (number % i == 0) {

count++;

}

}

return count;

}

void satisfyTheEquality(unsigned int k, unsigned int x, unsigned int y, int dx, int dy, unsigned int& count) {

if (k \* dx \* dy == x \* y) {

count++;

}

}

void yNumber(unsigned int n, unsigned int k, unsigned int x, unsigned int y,int dx, unsigned int& count) {

while (y <= n && x <= y) {

int dy = divisorsOfNumber(y);

satisfyTheEquality(k, x, y, dx, dy, count);

y++;

}

}

unsigned int countOfPairs(unsigned int n, unsigned int k) {

unsigned int x = 1, y = 1;

unsigned int count = 0;

while (x <= n && x <= y) {

int dx = divisorsOfNumber(x);

yNumber(n, k, x, y,dx, count);

x++;

y = x;

}

return count;

}

int main()

{

unsigned int n, k;

cin >> n >> k;

if (validationOfNumbers(n, k)) {

cout << countOfPairs(n, k);

}

else

cout << "Incorrect input";

}

/\*\*

\*

\* Solution to homework assignment 1

\* Introduction to programming course

\* Faculty of Mathematics and Informatics of Sofia University

\* Winter semester 2023/2024

\*

\* @author Petya Petrova

\* @idnumber 1MI0600311

\* @task 2

\*

\*/

#include <iostream>

using namespace std;

constexpr int SIZE = 1000;

bool validationOfN(unsigned int n) {

if (n < 1 || n > 1000) {

cout << "Incorrect input";

return 0;

}

return 1;

}

void readArray(int arr[], size\_t size) {

for (unsigned int i = 0; i < size; i++) {

unsigned int a;

cin >> a;

arr[i] = a;

}

}

int countOfNumber(int arr[], size\_t size, int& numberCounted, int position) {

int count = 0;

while (position < size) {

if (arr[position] == numberCounted) {

count++;

arr[position] = -1;

}

position++;

}

return count;

}

void groupingOfNumbers(int arr[], size\_t size, int& minPeople,int position) {

if (arr[position] != -1) {

int groups = 0;

int numberCounted = arr[position];

int count = countOfNumber(arr, size, numberCounted, position);

groups = count / (numberCounted + 1);

if (count % (numberCounted+1) != 0) {

groups++;

}

minPeople += groups \* (numberCounted + 1);

}

}

int minNumberOfPeopleInUpland(int arr[], size\_t size) {

int minPeople = 0;

for (int i = 0; i < size; i++) {

groupingOfNumbers(arr, size, minPeople,i);

}

return minPeople;

}

int main()

{

unsigned int n;

cin >> n;

if (validationOfN(n)) {

int arr[SIZE];

readArray(arr, n);

cout << minNumberOfPeopleInUpland(arr, n);

}

}

/\*\*

\*

\* Solution to homework assignment 1

\* Introduction to programming course

\* Faculty of Mathematics and Informatics of Sofia University

\* Winter semester 2023/2024

\*

\* @author Petya Petrova

\* @idnumber 1MI0600311

\* @task 3

\*

\*/

#include <iostream>

using namespace std;

constexpr int SIZE = 256;

void printArray(const char arr[], size\_t size) {

for (unsigned int i = 0; i < size; i++) {

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

}

}

void readArray(char arr[], size\_t size) {

for (unsigned int i = 0; i < size; i++) {

char a;

cin >> a;

arr[i] = a;

}

}

void removeAnElementsFromArray(char arr[], size\_t& size, unsigned int count, unsigned int indexGoingLeft, unsigned int indexGoingRight) {

indexGoingRight += 1;

for (unsigned int i = indexGoingLeft; indexGoingRight < size; i++, indexGoingRight++) {

arr[i] = arr[indexGoingRight];

}

size -= count;

}

bool checkIfThereIsRowBiggerThanThree(char arr[], size\_t& size, unsigned int count, unsigned int indexGoingLeft,

unsigned int indexGoingRight) {

if (count >= 3) {

removeAnElementsFromArray(arr, size, count, indexGoingLeft, indexGoingRight);

return 1;

}

return 0;

}

void countOfSymbol(char arr[], size\_t size, unsigned int& count, int index, int& indexLeft, int& indexRight) {

char symbol = arr[index];

count = 1;

indexLeft = index;

while (indexLeft != 0 && arr[indexLeft - 1] == arr[indexLeft])

{

indexLeft--;

count++;

}

indexRight = index;

while (indexRight != size - 1 && arr[indexRight + 1] == arr[indexRight])

{

indexRight++;

count++;

}

}

void checkForARow(char arr[], size\_t& size, unsigned int index) {

bool thereIsRowBiggerThanThree = false;

do {

int indexRight = index;

int indexLeft = index;

unsigned int count = 0;

countOfSymbol(arr, size, count, index, indexLeft, indexRight);

if (size == 0) break;

thereIsRowBiggerThanThree = checkIfThereIsRowBiggerThanThree(arr, size, count, indexLeft, indexRight);

index = indexLeft;

if (indexLeft > 0 && arr[indexLeft] != arr[indexLeft - 1]) break;

} while (thereIsRowBiggerThanThree);

}

void addingAnElementToTheArray(char arr[], size\_t& size, unsigned int index, char colour) {

arr[size] = colour;

for (unsigned int i = size - 1; i > index; i--) {

char temp = arr[i + 1];

arr[i + 1] = arr[i];

arr[i] = temp;

}

checkForARow(arr, size, index + 1);

}

void shoot(char arr[], size\_t& size, unsigned int shoots) {

unsigned int index;

char colour;

while (shoots != 0) {

cin >> index >> colour;

if (index <= size && size != 0) {

size++;

addingAnElementToTheArray(arr, size, index, colour);

}

shoots--;

}

}

int main()

{

size\_t numberOfBalls;

cin >> numberOfBalls;

char arr[SIZE];

readArray(arr, numberOfBalls);

unsigned int shoots;

cin >> shoots;

shoot(arr, numberOfBalls, shoots);

if (numberOfBalls == 0)

cout << -1;

else

printArray(arr, numberOfBalls);

}

/\*\*

\*

\* Solution to homework assignment 1

\* Introduction to programming course

\* Faculty of Mathematics and Informatics of Sofia University

\* Winter semester 2023/2024

\*

\* @author Petya Petrova

\* @idnumber 1MI0600311

\* @task 4

\*

\*/

#include <iostream>

using namespace std;

bool verificationOfInput(unsigned int yearBefore, unsigned int monthBefore, unsigned int dateBefore,

unsigned int yearAfter, unsigned int monthAfter, unsigned int dateAfter,

unsigned int interval) {

if (interval < 1 || interval>365) {

return 0;

}

if (yearBefore > yearAfter || yearBefore < 1600 || yearAfter>2100) {

return 0;

}

if (monthBefore > 12 || monthBefore < 1 || monthAfter>12 || monthAfter < 1) {

return 0;

}

if (dateBefore > 31 || dateAfter > 31 || dateBefore < 1 || dateAfter < 1) {

return 0;

}

if (yearBefore == 1916 && monthBefore == 4 && dateBefore >= 1 && dateBefore <= 13) {

return 0;

}

if (yearAfter == 1916 && monthAfter == 4 && dateAfter >= 1 && dateAfter <= 13) {

return 0;

}

return 1;

}

void newStyle(unsigned int& year, unsigned int& month, unsigned int& date) {

if (year % 4 == 0 && year % 100 != 0) {

if (date >= 30) {

date -= 29;

month++;

}

}

else if (year % 400 == 0) {

if (date >= 30) {

date -= 29;

month++;

}

}

else if (date >= 29) {

date -= 28;

month++;

}

}

void oldStyle(unsigned int& year, unsigned int& month, unsigned int& date) {

if (year % 4 == 0) {

if (date >= 30) {

date -= 29;

month++;

}

}

else {

date -= 28;

month++;

}

}

void february(unsigned int& year, unsigned int& month, unsigned int& date) {

if (year <= 1916) {

oldStyle(year, month, date);

}

else

newStyle(year, month, date);

}

void April1916(unsigned int currentYear, unsigned int currentMonth, unsigned int& currentDate, unsigned int yBefore, unsigned int mBefore, unsigned int dBefore) {

bool hasNotPassedYet = true;

if (currentYear == 1916 && mBefore < 4) {

currentDate += 13;

hasNotPassedYet = false;

}

}

void dateBiggerThenTheMonth(unsigned int& currentYear, unsigned int& currentMonth, unsigned int& currentDate, unsigned int yBefore, unsigned int mBefore, unsigned int dBefore) {

while (currentDate > 31) {

switch (currentMonth) {

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:currentDate -= 31;

currentMonth++; break;

case 4:April1916(currentYear, currentMonth, currentDate, yBefore, mBefore, dBefore);

case 6:

case 9:

case 11:currentDate -= 30;

currentMonth++; break;

case 2:

february(currentYear, currentMonth, currentDate); continue;

case 12:currentDate -= 31;

currentMonth = 1;

currentYear++; break;

}

}

}

void validationOfCurrentDate(unsigned int currentYear, unsigned int currentMonth, unsigned int& currentDate) {

if (currentYear == 1916 && currentMonth == 4 && currentDate == 1) {

currentDate = 14;

}

}

void calendar(unsigned int yBefore, unsigned int mBefore, unsigned int dBefore, unsigned int yAfter,

unsigned int mAfter, unsigned int dAfter, unsigned int i) {

unsigned int currentYear = yBefore;

unsigned int currentMonth = mBefore;

unsigned int currentDate = dBefore;

while (!(currentYear >= yAfter && currentMonth >= mAfter && currentDate >= dAfter)) {

validationOfCurrentDate(currentYear, currentMonth, currentDate);

cout << currentYear << " " << currentMonth << " " << currentDate << " - ";

currentDate += i - 1;

if (currentYear == yAfter && currentMonth == mAfter && currentDate > dAfter) break;

dateBiggerThenTheMonth(currentYear, currentMonth, currentDate, yBefore, mBefore, dBefore);

switch (currentMonth) {

case 4:April1916(currentYear, currentMonth, currentDate, yBefore, mBefore, dBefore);

case 6:

case 9:

case 11:

if (currentDate == 31) {

currentDate -= 30;

currentMonth++;

}break;

case 2:

february(currentYear, currentMonth, currentDate);

break;

}

if (currentYear >= yAfter && currentMonth >= mAfter && currentDate >= dAfter) {

break;

}

else {

cout << " " << currentYear << " " << currentMonth << " " << currentDate << endl;

}

currentDate++;

dateBiggerThenTheMonth(currentYear, currentMonth, currentDate, yBefore, mBefore, dBefore);

}

if (currentYear == yAfter && currentMonth == mAfter && currentDate == dAfter) {

cout << currentYear << " " << currentMonth << " " << currentDate << " - " << yAfter << " " << mAfter << " " << dAfter;

}

else

cout << yAfter << " " << mAfter << " " << dAfter;

}

int main()

{

unsigned int yearBefore, monthBefore, dateBefore;

unsigned int yearAfter, monthAfter, dateAfter;

unsigned int interval;

cin >> yearBefore >> monthBefore >> dateBefore >> yearAfter >> monthAfter >> dateAfter >> interval;

if (!verificationOfInput(yearBefore, monthBefore, dateBefore, yearAfter, monthAfter, dateAfter, interval)) {

cout << "Incorrect input";

}

calendar(yearBefore, monthBefore, dateBefore, yearAfter, monthAfter, dateAfter, interval);

}