**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им.В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем

Лабораторная работа №2.2

дисциплина: Дискретная математика

тема: «Задачи выбора»

Выполнил: ст. группы ПВ-201

Машуров Дмитрий Русланович

Проверил: Бондаренко Т.В.

Белгород 2021

**Лабораторная работа № 2.2**

**Задачи выбора**

**Цель работы:** приобрести практические навыки в использовании алгоритмов порождения комбинаторных объектов при проектировании алгоритмов решения задач выбора.

**Задания**

1. Ознакомиться с задачей (см. варианты заданий).
2. Определить класс комбинаторных объектов, содержащих решение задачи (траекторию задачи).
3. Определить, что в задаче является функционалом и способ его вычисления.
4. Определить способ распознавания решения по значению функционала.
5. Реализовать алгоритм решения задачи.
6. Подготовить тестовые данные и решить задачу.

**Задание варианта №17**

Задана целочисленная матрица размером n × m. Выбрать минимальное количество строк матрицы, таких, что сумма элементов каждого столбца была бы больше заданного числа.

**Выполнение:**

1. Определю класс комбинаторных объектов, содержащих решение задачи (траекторию задачи):

Исходя из условия задачи, мне необходимо получить различные комбинации строк матрицы, которые будут участвовать в вычислении минимального значения

Таким образом классом комбинаторных объектов будут являться подмножества множества строк матрицы

1. Определю, что в задаче является функционалом и способ его вычисления

Функционалом будет являться двоичный вектор , определяющий подмножества множества строк матрицы

Вычисляться он будет с помощью переборного алгоритма порождения подмножеств множества

1. Определю способ распознавания решения по значению функционала:

Двоичный вектор (функционал), определяющий некоторое подмножество множества строк матрицы, будет являться решением, если сумма каждого столбца данного подмножества строк будет больше заданного числа , а количество строк при этом будет минимальным

1. Реализую алгоритм решения задачи:

#include <stdio.h>  
  
#define N 100  
  
int MIN\_ROWS = N, num, ROWS\_MASK[N], a[N][N];  
  
/\*вывод матрицы с маской row\_mask\*/  
void print\_matr\_with\_row\_mask(const int row\_mask[], int m, int n) {  
 for (int i = 0; i < m; ++i) {  
 if (row\_mask[i]) {  
 for (int j = 0; j < n; ++j) {  
 printf("%d ", a[i][j]);  
 }  
  
 printf("\n");  
 }  
 }  
}  
  
/\*копирование массива this размера n в массив other\*/  
void copy\_arr(const int this[], int n, int other[]) {  
 for (int i = 0; i < n; ++i) {  
 other[i] = this[i];  
 }  
}  
  
/\*возвращает сумму столбца col по выбранной маске строк mask длины m\*/  
int get\_col\_sum\_by\_mask(const int \*mask, int m, int col) {  
 int sum = 0;  
  
 for (int i = 0; i < m; ++i) {  
 if (mask[i] == 1) {  
 int t = a[i][col];  
 sum += t;  
 }  
 }  
  
 return sum;  
}  
  
/\*возвращает "истину", если каждое значение массива cols\_sum размера n  
 \*превышает num, иначе - "ложь"\*/  
static int check\_cols\_sum(int n, const int \*cols\_sum) {  
 for (int j = 0; j < n; ++j) {  
 if (cols\_sum[j] <= num) {  
 return 0;  
 }  
 }  
  
 return 1;  
}  
  
/\*помещает сумму столбцов в массив cols\_sum\*/  
static void set\_cols\_sum(int m, int n, const int \*d, int \*cols\_sum) {  
 for (int j = 0; j < n; ++j) {  
 int sum = get\_col\_sum\_by\_mask(d, m, j);  
 cols\_sum[j] = sum;  
 }  
}  
  
/\*рекурсивная функция: генерирует подмножества множества строк матрицы a  
 \*размера m строк и n столбцов\*/  
static void get\_subsets\_inner(int m, int n, int i) {  
 static int d[100];  
 static int cols\_sum[N];  
  
 for (int x = 0; x < 2; ++x) {  
 d[i] = x;  
  
 if (i == n - 1) {  
 set\_cols\_sum(m, n, d, cols\_sum);  
  
 int c = 0;  
 for (int j = 0; j < n; ++j) {  
 if (d[j] == 1) c++;  
 }  
  
 if (check\_cols\_sum(n, cols\_sum) && c < MIN\_ROWS) {  
 MIN\_ROWS = c;  
 copy\_arr(d, n, ROWS\_MASK);  
 }  
 } else {  
 get\_subsets\_inner(m, n, i + 1);  
 }  
 }  
}  
  
/\*функция-оболочка для get\_subsets\_inner\*/  
void get\_rows\_subsets(int m, int n) {  
 get\_subsets\_inner(m, n, 0);  
}  
  
int main() {  
 scanf("%d", &num);  
  
 int n, m;  
 scanf("%d %d", &m, &n);  
  
 for (int i = 0; i < m; ++i) {  
 for (int j = 0; j < n; ++j) {  
 scanf("%d", &a[i][j]);  
 }  
 }  
  
 get\_rows\_subsets(m, n);  
 printf("%d\n", MIN\_ROWS);  
 print\_matr\_with\_row\_mask(ROWS\_MASK, m, n);  
  
}

Тестовые данные:

|  |  |  |
| --- | --- | --- |
| № | Вход | Выход |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |