题目描述：

一个有 N 个选手参加比赛，选手编号为 1~N（3<=N<=100），有 M（3<=M<=10）个评委对选手进行打分。打分规则为每个评委对选手打分，最高分 10 分，最低分 1 分。

请计算得分最多的 3 位选手的编号。如果得分相同，则得分高分值最多的选手排名靠前(10 分数量相同，则比较 9 分的数量，以此类推，用例中不会出现多个选手得分完全相同的情况)。

输入描述：

第一行为半角逗号分割的两个正整数，第一个数字表示 M（3<=M<=10）个评委，第二个数字表示 N（3<=N<=100）个选手。

第 2 到 M+1 行是半角逗号分割的整数序列，表示评委为每个选手的打分，0 号下标数字表示 1 号选手分数，1 号下标数字表示 2 号选手分数，依次类推。

输出描述：

选手前 3 名的编号。

注：若输入为异常，输出-1，如 M、N、打分不在范围内。

示例 1

输入：

```
4,5

10,6,9,7,6
```

```
9,10,6,7,5

8,10,6,5,10

9,10,8,4,9
```

输出：

```
2,1,5
```

说明：

第一行代表有 4 个评委，5 个选手参加比赛

矩阵代表是 4*5，每个数字是选手的编号，每一行代表一个评委对选手的打分排序，

2 号选手得分 36 分排第 1，1 号选手 36 分排第 2，5 号选手 30 分(2 号 10 分值有 3 个，1 号 10 分值只有 1 个，所以 2 号排第一)

示例 2

输入：

```
2,5

7,3,5,4,2

8,5,4,4,3
```

输出：

```
-1
```

说明：

只有 2 个评委，要求最少为 3 个评委

示例 3

输入：

```
4,2

8,5

5,6

10,4

8,9
```

输出：

```
-1
```

说明：

只有 2 名选手参加，要求最少为 3 名

示例 4

输入：

```
4,5
11,6,9,7,8
9,10,6,7,8
8,10,6,9,7
9,10,8,6,7
```

输出：

```
-1
```

说明：

第一个评委给第一个选手打分 11，无效分数

```
#include "stdio.h"
#include "stdlib.h"
#include "string.h"

typedef struct asf{
	int allscore;
	int num;
}all;
```

```
int cmp(const void *a, const void *b)
{
	return *(int*)b-*(int*)a;
}

int cmps(const void *a, const void *b)
{
	return (*(all*)b).allscore-(*(all*)a).allscore;
}

int main() {
	int i, j, k, m, n, temp, flag = 0;
	while (scanf("%d,%d", &m, &n) != EOF) {

		if ((m > 10) || (m < 3)) {
			die:printf("-1");
			break;
		}
		if ((n > 100) || (n < 3)) {
			printf("-1");
			break;
		}

		int score[n][m];
		all allscore[n];
		scanf("\n");
		for (i = 0; i < m; i++)
		{
			for (j = 0; j < n; j++)
			{
				scanf("%d,", &score[j][i]);
				//printf("%d,",score[j][i]);
				if ((score[j][i] > 10) || (score[j][i] < 1))
				{
					flag = 1;
				}
			}
			//printf("\n");
		}

		if(flag)
		{
			printf("-1");
			break;
```

```
}

for (j = 0; j < n; j++)
{
	qsort(score[j], m, sizeof(int), cmp);
	allscore[j].allscore = 0;
	allscore[j].num = j+1;
	for (i = 0; i < m; i++)
	{
		allscore[j].allscore += score[j][i];
	}
	//printf("%d ",allscore[j].allscore);
}

qsort(allscore, n, sizeof(all), cmps);

//for (k = 0; k < n-1; k++)
for (i = 0; i < n-1; i++)
{
	if(allscore[i].allscore == allscore[i+1].allscore)
		for(j = 0; j < m; j++)
		{
			if(score[allscore[i].num - 1][j] > score[allscore[i+1].num - 1][j])
				break;
			else if(score[allscore[i].num - 1][j] == score[allscore[i+1].num - 1][j])
				continue;
			else if(score[allscore[i].num - 1][j] < score[allscore[i+1].num - 1][j])
			{
				temp = allscore[i].num;
				allscore[i].num = allscore[i+1].num;
				allscore[i+1].num = temp;

				temp = allscore[i].allscore;
				allscore[i].allscore = allscore[i+1].num;
				allscore[i+1].allscore = temp;

				break;
			}
		}
	//printf("%d,",allscore[i].num);
	//printf("%d    ",allscore[i].allscore);
}
```

```
            for (i = 0; i < 2; i++)
            {
                printf("%d,",allscore[i].num);
                //printf("%d    ",allscore[i].allscore);
            }
            printf("%d",allscore[i].num);
            //printf(",%d",allscore[i].allscore);
        }
        return 0;
}
```