# 3. ЗАКЛЮЧИТЕЛЬНЫЙ ЭТАП

# Предметный тур

## 4.1. Математика. 9 класс

### *Задача 4.1.1. (15 баллов)*

Найдите общий корень уравнений

$$x^2 - x - 42 = 0,$$
$$3x^3 - 16x^2 - 33x - 14 = 0.$$

#### *Решение*

Решив первое уравнение каким-либо известным способом находим его корни

$$x_1 = 7, x_2 = -6.$$

Аналогично находим корни второго уравнения:

$$x_1 = -1, x_2 = 7, x_3 = -\frac{2}{3}.$$

Общим корнем будет $x = 7$.

**Ответ:** 7.

**Критерии оценки**

+4 баллов за нахождение корней первого уравнения;

+9 баллов за нахождение корней второго уравнения;

+2 балла за выбор общего корня.

<u>Замечания</u>

1. За нахождения общего корня перебором баллы снижаются только в случае, когда не была проведена проверка того, что найденное число является корнем <u>обоих</u> уравнений.
2. Если найденный общий корень не совпадает с ответом, то за задачу ставится 0 баллов.
3. Если при вычислении корней обоих уравнений были совершены арифметические ошибки, то баллы за каждое из этих уравнений ставятся пропорционально числу правильно найденных корней.

## *Задача 4.1.2. (15 баллов)*

Найдите все простые числа $p$, для которых $2p^4+46$ — квадрат целого числа.

### *Решение*

Найдем для начала хотя бы одно решение. Для этого переберем малые простые числа и убедимся, что $p=5$ подходит.

Покажем, что других решений у уравнения нет. Для этого заметим, что если $p \neq 5$, то оно взаимно просто с пятью, а значит $p^4$ дает остаток 1 при делении на 5.

Тогда выражение $2p^4+46$ дает остаток 3 при делении на 5. Легко убедиться, что квадраты целых чисел могут давать только остатки $0, 1, 4$. Следовательно, больше нет решений в целых числах.

**Ответ:** $p=5$.

**Критерии оценки**

+5 баллов за подобранный пример;

+10 баллов за доказательство единственности.

## *Задача 4.1.3. (20 баллов)*

Дана трапеция с основаниями $AB = 18\sqrt{3}$ и $CD = 12\sqrt{3}$. На $AB$ и $CD$ взяты точки $M$ и $N$ соответственно так, что $MN = 6$.

а) *(5 баллов)* Найдите площадь трапеции, если дополнительно известно, что угол между $MN$ и $AB$ равен $60°$.

б) *(15 баллов)* Найдите наибольшее возможное значение наименьшего из углов трапеции, если угол $A$ — прямой.

### *Решение*

а) Заметим, что высота в такой трапеции будет равна $h = \sin 60° \cdot 6 = 3\sqrt{3}$. Тогда по формуле площади трапеции $S = \frac{1}{2}h(a+b) = 135$.

б) Так как $AB > CD$, то острый угол $B$ — наименьший из углов трапеции. Опустим из вершины $C$ перпендикуляр $CE$ на $AB$. Его длина не больше длины $MN = 6$. Кроме того, из условия следует, что

$$BE = 18\sqrt{3} - 12\sqrt{3} = 6\sqrt{3}.$$

Поэтому $\operatorname{tg} \angle B \leqslant \dfrac{1}{\sqrt{3}}$, откуда $\angle B \leqslant 30°$.

Для того чтобы получить максимальное значение угла $B$, достаточно предположить, что $MN$ — высота трапеции.

**Ответ:** а) 135.

**Критерии оценки**

а) *(5 баллов)*
+5 баллов за правильно найденную площадь.

б) *(15 баллов)*
+10 баллов за доказанную оценку;
+5 баллов за правильно подобранный пример с углами 30° и 150°.

## *Задача 4.1.4. (25 баллов)*

На аукционе Илья соревнуется с Петей за ценную картину. Изначальная ее цена — 1000 рублей. За раз стоимость лота можно поднять на 2000 или 5000 рублей. Парни так сильно жаждут получить картину, что не уйдут, пока она не достанется одному из них. Аукционист сообщил, что если цена за картину поднимется выше 28200 рублей, то последний поднявший цену получит картину. Первым цену поднял Илья... (на сколько — вы не знаете)

а) *(10 баллов)* Правильно ли он поступил? Сможет ли он при любом поведении его соперника, получить заветный лот?

б) *(15 баллов)* Изменится ли ответ, если вместо 2000 или 5000 рублей ребята могут поднимать цену на 20 или 50?

### *Решение*

а) Покажем, что Петя может всегда выиграть, как бы Илья не ходил. Для этого Петя может всегда ходить противоположнвм образом, то есть если Илья поднял цену на 2000 рублей, то Петя поднимет на 5000 рублей и наоборот. Тогда после 4 таких ходов стоимость станет равной 29000 рублей и Петя заберет картину.

б) Рассмотрим игру с конца с позиции выигрышных и проигрышных позиций. Так как максимальная цена — 28200 рублей, то 28200 — выигрышная, 28190 — выигрышная, ..., 28160 — выигрышная, 28150 — проигрышная, 28140 — проигрышная, 28130 — выигрышная, 28120 — выигрышная, 28110 — проигрышная, 28100 — выигрышная, 28090 — выигрышная, 28080 — проигрышная, 28070 — проигрышная, ...

Можно заметить, что в данном случае будут повторяться постоянно 7 позиций:

$$\ldots \text{П, П, В, В, П, В, В} \ldots$$

Итак, получим, что 1050 и 1020 — это проигрышные позиции, поэтому как бы первый раз не сходил Илья он может потом продолжить играть так, чтобы выиграть.

**Ответ:** а) Нет, у Пети есть выигрышная стратегия; б) Да, у Ильи есть выигрышная стратегия.

**Критерии оценки**

а) *(10 баллов)*

+5 баллов за нахождение стратегии для Пети;

+5 баллов за рассуждения о том, что Илья проиграет, независимо от своей игры.

б) *(15 баллов)*

+10 баллов за доказательство того факта, что позиции повторяются;

+5 баллов за рассуждения о том, что Илья выиграет, независимо от игры оппонента.

Замечания

1. Если без каких-либо объяснений сказано, что позиции повторяются, то ставится 5 вместо 10 баллов.
2. Если правильно найден вид периода позиций, но нет рассуждений, почему, если рассматривать с конца, будут повторяться именно эти 7 позиций, то ставится аналогично 5 вместо 10 баллов.
3. Если все рассуждения о периодичности сделаны верно, но для позиций 1050 и 1020 сказано, что хотя бы одна выигрышная, то снимается все 5 баллов.

## *Задача 4.1.5. (25 баллов)*

Вася случайным образом поставил на доску

а) *(10 баллов)* двух королей;

б) *(15 баллов)* короля, ладью и ферзя;

С какой вероятностью они будут бить друг друга (во втором пункте подразумевается, что каждые две фигуры будут бить друг друга)?

Замечание: король бьет 8 клеток вокруг себя; ладья бьет те фигуры, которые стоят с ней в одном ряду или столбце (ладья не может перепрыгивать через фигуры); ферзь бьет те фигуры, которые находят с ним в одном ряду или столбце или на одной диагонали (аналогично ферзь не может перепрыгивать через фигуры).

### *Решение*

а) Всего вариантов расставить двух различных королей — $64 \cdot 63 = 4032$. Теперь рассмотрим варианты, когда они бьют друг друга. Таких вариантов $4 \cdot 3 + 24 \cdot 5 + 36 \cdot 8 = 420$. Искомая вероятность будет равна $\frac{420}{4032} = \frac{5}{48}$

б) Эти три фигуры могут попарно друг друга бить только в том случае, если они стоят «уголком», где король и ферзь стоят по диагонали, а ладья соседствует с ними обоими. Тогда заметим, что всего подобных уголков в каждом $2 \times 2$ квадратике можно сделать 8 штук, а значит, так как в $8 \times 8$ есть 49 различных (они накладываются друг на друга) квадратов $2 \times 2$, то всего вариантов, когда все они друг друга бьют — $49 \cdot 8 = 392$.

Следовательно, так как всего $64 \cdot 63 \cdot 62 = 249984$ вариантов, то искомая вероятность равна $\frac{392}{249984} = \frac{7}{4464}$.

**Ответ:** а) $\frac{5}{48}$; б) $\frac{7}{4464}$.

**Критерии оценки**

а) *(10 баллов)*

+10 баллов за рассуждения, объясняющие найденную вероятность.

б) *(15 баллов)*

+5 баллов за рассуждения о возможных позициях фигур;

+10 баллов за рассуждения, объясняющие найденную вероятность.

Замечания

1. Если в процессе вычислений вероятностей допущены арифметические ошибки, но все рассуждения правильны, то балл не снижается.

## 4.2. Математика. 10-11 класс

### *Задача 4.2.1. (10 баллов)*

Дано 8 чисел, одно из которых целое. Известно, что сумма любых семи из них целая. Могут ли среди чисел быть нецелые?

***Решение***

Рассмотрим сумму без и с целым числом и вычтем одно из другого. Тогда получим, что $x - a = b$, где $a$ и $b$ — это какие-то целые числа. Тогда $x$ тоже целое число. В силу произвольности $x$ следует, что все восемь чисел целые.

**Ответ:** Нет, не могут.

**Критерии оценки**

+10 баллов за рассуждения о том, почему все числа целые.

### *Задача 4.2.2. (20 баллов)*

В двух различных узлах бесконечной сетки сидят заяц и волк. Они оба могут перемещаться только по сторонам клеток. Первым двигается заяц, потом волк, после него заяц... За один свой ход волк преодолевает $v$ сторон клеток, а заяц — $z$ (двигаются они строго по сторонам клеток).

а) *(10 баллов)* Пусть скорости у них равны. Докажите, что заяц может так действовать, чтобы волк его никогда не догнал.

б) *(10 баллов)* Пусть скорость волка $v = z + 1$. Докажите, что при любом расположении персонажей волк сможет догнать зайца.

***Решение***

а) Пусть заяц всегда будет двигаться от волка. Заметим, что он всегда может двигаться так, чтобы кратчайшее расстояние между ним и волком увеличилось ровно на $z$. Тогда, как бы волк ни двигался, но после его хода расстояние между ними не сократится более, чем на $v = z$. Поэтому за два хода (зайца и волка) расстояние между ними не уменьшится, а следовательно, если оно изначально было положительным, то и после любого хода будет оставаться таким же.

б) В этом пункте нам надо смотреть на погоню со стороны волка. Он может повторить все движения зайца и при этом еще пройти на одну клетку больше. Тогда пусть он будет последним передвижением сокращать расстояние между ними. При таком поведении расстояние между волком и зайцем будет сокращаться всегда хотя бы на единицу после их двух ходов. Так как в начале игры оно было конечным, то наступит момент, когда волк догонит зайца.

**Критерии оценки**

а) *(10 баллов)*

+10 баллов за рассуждения о том, почему после двух ходов заяц сможет не уменьшить расстояние между ним и волком.

б) *(10 баллов)*

+10 баллов за рассуждения о том, почему волк может сокращать расстояния между ним и зайцем.

<u>Замечания</u>

1. Может быть приведено более оптимальное решение, где волк не повторяет движения зайца, а просто сокращает расстояния между ними, двигаясь по кратчайшему пути. В этом случае, если не приведено обоснований, почему после двух ходов расстояние между ними сократится, то ставится половина баллов.
2. За решения без обоснований и пояснений, почему волк догонит или не догонит зайца, оцениваются нулем баллов.

## *Задача 4.2.3. (20 баллов)*

На доске написано $k$ различных ненулевых действительных чисел. Известно, что если выбрать $m$ ($1 \leqslant m \leqslant k$) произвольных различных чисел с доски и перемножить, то получим число с доски.

а) *(15 баллов)* Найдите максимальное возможное значение $k$.

б) *(5 баллов)* Дополнительно известно, что все числа по модулю не превосходят 2019. Чему равняется максимально возможная сумма чисел с доски? Приведите пример, когда это значение суммы достигается

***Решение***

а) Во-первых, покажем, что у нас есть только одно число, модуль которого больше 1. Если бы таких чисел было бы несколько, то можно было бы выбрать два числа $a, b$ с наибольшими модулями и тогда их произведение было бы больше любого из чисел на доске, что противоречит условию. Аналогичные рассуждения можно провести для чисел, меньших единицы по модулю.

Во-вторых, если у нас есть число не равное $\pm 1$, то $-1$ не может входить в набор. Иначе бы мы получили пару чисел больших единицы по модулю или меньших и вернулись бы к предыдущим двум рассуждениям. Следовательно, всего чисел не больше трех. Пример: $\frac{1}{2}, 1, 2$.

б) Заметим, что всевозможные наборы чисел, которые могут быть выписаны на доске, выглядят так: $\frac{1}{n}, 1, n$ (или их какое-то подмножество). Следовательно, максимально возможная сумма чисел будет достигаться при $n = 2019$ (можно показать, что при увеличении $n$ сумма всех трех чисел будет увеличиваться: например, взяв производную от суммы, как функции, зависящей от $n$).

**Ответ:** б) $2020\frac{1}{2019}$.

### Критерии оценки

а) *(15 баллов)*

+5 баллов за рассуждения о том, почему нет больше 1 числа, большего единицы.

+5 баллов за рассуждения о том, почему нет больше 1 числа, меньшего единицы.

+5 баллов за рассуждения о том, почему нет $-1$ в наборе.

б) *(5 баллов)*

+3 балла за рассуждения о том, почему подходящий набор будет составлен из трех чисел: единицы, числа $n$ и его обратного.

+1 балл за рассуждения, почему при увеличении $n$ сумма увеличивается.

+1 балл за пример.

## *Задача 4.2.4. (25 баллов)*

Перед Васей и Петей стоят три корзины (желтая, белая и красная) и лежат 17 пронумерованных шаров. Сначала Вася берет по шару и с равной вероятностью кладет в одну из корзин. Затем Петя достает шар с номером 1 из той корзины, в которой он лежит.

а) *(5 баллов)* Какова вероятность того, что после этого среди корзин будет хотя бы одна пустая?

б) *(20 баллов)* После этого Петя с вероятностью $\frac{1}{2}$ достает из корзин шар с номером 2, или с вероятностью $\frac{1}{4}$ он достает из корзин два шара: с номером

2 и 3, или с вероятностью $\frac{1}{8}$ он достает из корзин три шара: с номером 2, 3 и 4 ... Все 16 оставшихся шаров он достает с вероятностью $\frac{1}{2^{15}}$.

Какова вероятность того, что после этого среди корзин будет хотя бы одна пустая? (в этом пункте ответ может быть получен в виде суммы некоторого выражения; при его получении дальнейшие сокращения можно не проводить)

***Решение***

а) Допустим, что всего у нас $n$ шаров ($n > 2$) в 3-х корзинах. Посчитаем вероятность того, что все корзины не пустые, с помощью формулы включений-исключений

$$p = 3 \cdot \left(\frac{2}{3}\right)^n - 3 \cdot \left(\frac{1}{3}\right)^n.$$

Заметим, что в наших корзинах остается 16 определенных шаров всегда, поэтому в этом пункте ответ будет получаться подстановкой $n = 16$ в формулу выше.

б) Здесь у нас повторяется тоже, что и в предыдущем пункте, но $n$ может принимать разные значения с разной вероятностью. Тогда с вероятностью $\frac{1}{2}$ останется $n = 15$ шаров, с вероятностью $\frac{1}{4}$ останется $n = 14$ шаров, ... Тогда итоговой вероятностью будет в каждом случае произведение вероятности того, что все три корзины не пустые при определенном числе шаров и вероятности того, что после выбора Пети останется это число шаров:

$$P = \frac{1}{2^{15}} + \frac{1}{2^{15}} + \frac{1}{2^{14}} + \sum_{n=3}^{15} \left(3 \cdot \left(\frac{2}{3}\right)^n - 3 \cdot \left(\frac{1}{3}\right)^n\right) \cdot \frac{1}{2^{16-n}}.$$

В этой сумме первые три слагаемых — это случаи, когда останется 0, 1 и 2 шара. Это сумму можно было не сокращать, хотя имеет место следующие выкладки.

Сократим немного выражение:

$$P = \frac{1}{2^{13}} + 3 \cdot \frac{1}{2^{13}} \cdot \sum_{t=0}^{12} \left( \cdot \left(\frac{2}{3}\right)^{t+3} - 3 \cdot \left(\frac{1}{3}\right)^{t+3}\right) \cdot 2^t.$$

По формуле суммы геометрической прогрессии имеем

$$P = \frac{1}{2^{13}} + \frac{3}{2^{13}} \cdot \left\{ \left(\frac{2}{3}\right)^3 \cdot \frac{1 - \left(\frac{2}{3}\right)^{13}}{1 - \frac{2}{3}} + \left(\frac{1}{3}\right)^3 \cdot \frac{1 - \left(\frac{1}{3}\right)^{13}}{\frac{1}{3} - 1} \right\}$$

**Ответ:** а), б) см. решение.

**Критерии оценки**

а) *(5 баллов)*

+2 балла за рассуждения, почему можно рассматривать 16 шаров вместо 17.

+3 балла за использование формулы включений-исключений.

б) *(20 баллов)*

+5 баллов за рассуждения о том, почему, как и в первом пункте, в каждом случае у нас вероятность зависит только от числа оставшихся шаров.

+5 баллов за рассуждения о том, почему надо перемножать вероятности.

+10 баллов за правильно выписанную формулу и объяснения, почему она так выглядит (включая вынесенные первые слагаемые).

<u>Замечания</u>

1. Если во втором пункте не разобран случай с $n = 0, 1, 2$, т. е. сумма итоговая неправильна и нет указаний нигде о том, что в этих случаях будет, то балл за формулу снижается на 5 баллов.

## *Задача 4.2.5. (25 баллов)*

В прямоугольном треугольнике $ABC$ (с прямым углом $B$) отметили точки $D$ и $E$ на сторонах $AB$ и $BC$ так, что $DE$ параллельно $AC$. Из угла $B$ опустили высоту $BF$ на $AC$ и на ней отметили точку $L$, из которой отрезок $AD$ виден под прямым углом. Докажите, что лучи $AL$ и $LD$ отсекают на прямой $BC$ отрезок, равный $EC$.

### *Решение*

Обозначим $\angle FBC = \alpha$. Тогда можно показать, что $AD \cdot \operatorname{tg}\alpha = EC$ (можно, например, опустить высоты из $E$ и $D$ на $AC$ и приравнять их, воспользовавшись тем, что $\angle CAB = \alpha$).

Пусть точки $M$ и $N$ получаются пересечением лучей $AL$ и $LD$ с прямой $BC$. Соединим точки $A$ и $N$. Тогда $D$ будет ортоцентром треугольника $AMN$.

Так как $AB$ и $NL$ — высоты в треугольнике $AMN$, то $\angle LBM = \angle MAN = \alpha$. Кроме того, равны углы $NMA$ и $ADL$. Поэтому треугольники $ADL$ и $NLM$ подобны, откуда

$$\frac{AD}{MN} = \frac{AL}{LN} = \operatorname{ctg}\angle MAN = \operatorname{ctg}\alpha \Rightarrow AD\operatorname{tg}\alpha = MN.$$

Из двух равенств выше имеем искомое $MN = EC$.

### Критерии оценки

+10 баллов за вывод формулы $AD \cdot \operatorname{tg}\alpha = EC$.

+5 баллов за замечание, что $D$ ортоцентр треугольника $AMN$.

+10 баллов за вывод формулы $AD\operatorname{tg}\alpha = MN$.

<u>Замечания</u>

1. За полностью правильное решение, которое отличается от авторского баллы не снижать (в случае наличия ошибок, снижать баллы пропорционально их тяжести и влияния на решение задачи).

# 4.3. Информатика

## *Задача 4.3.1. Подарок на 14 февраля (10 баллов)*

Саша и Лена живут в противоположных домах. Каждый вечер в каждом из этих домов загорается свет в некоторых окнах, в результате чего живущие напротив могут наблюдать некоторый узор из окон, в которых горит свет. Окна в домах Лены и Саши устроены стандартным образом — они образуют прямоугольную клетчатую сетку $N$ строк на $M$ столбцов.

Саша знает любимый узор Лены и хочет порадовать её. Для этого он может пойти к соседям по дому и попросить у них выключить свет на вечер. Саша хочет, чтобы в результате этого окна, оставшиеся гореть, образовывали любимый узор Лены и ничего более.

Однако Саша подозревает, что он может сделать это не единственным образом. Ему стало интересно, сколькими способами он может попросить соседей выключить свет (обращаем ваше внимание на то, что просить соседей включать свет он не может — тому есть свои причины), так чтобы горящие окна его дома обрадовали Лену. Порядок, в котором выключаются окна, неважен.

Поскольку вечер уже начался и ему пора начать обход соседей, Саша просит вас помочь ему с подсчётом.

### Формат входных данных

В первой строке вводятся 2 целых числа $N, M (1 \leqslant N, M \leqslant 10)$ — число этажей и число окон на каждом этаже в доме Саши. $i$-я из следующих $N$ строк содержит $M$ чисел (1 или 0) — горит ли соответствующее окно на $(N - i + 1)$-м этаже или нет соответственно.

В $N+2$-й строке вводятся 2 целых числа $H, W (1 \leqslant H, W \leqslant 10)$ — высота и ширина любимого узора Лены. $i$-я из следующих $H$ строк содержит $W$ чисел (1 или 0) — цвет соответствующей ячейки в $(N-i+1)$-м ряду в любимом узоре Лены соответственно.

### Формат выходных данных

Выведите в ответ одно целое число — количество способов, которыми Саша может попросить соседей выключить свет, так чтобы оставшиеся гореть окна образовывали только любимый узор Лены.

#### *Пояснения к ответу*

В первом тесте любимый узор Лены можно нарисовать в любом месте Сашиного дома, всего таких вариантов $(N - 1) \cdot (M - 1) = 4 \cdot 5 = 20$.

*Пример №1*

| Стандартный ввод |
|---|
| 5 6<br>1 1 1 1 1 1<br>1 1 1 1 1 1<br>1 1 1 1 1 1<br>1 1 1 1 1 1<br>1 1 1 1 1 1<br>2 2<br>1 0<br>0 1 |

| Стандартный вывод |
|---|
| 20 |

*Пример №2*

| Стандартный ввод |
|---|
| 5 5<br>1 1 0 0 0<br>1 1 0 0 0<br>0 0 0 0 0<br>0 0 0 0 0<br>0 0 0 0 0<br>2 2<br>1 0<br>0 0 |

| Стандартный вывод |
|---|
| 4 |

***Решение***

Переберем все возможные позиции, в которых можно оставить любимый узор Лены. Заметим, что позиция подходит тогда и только тогда, когда каждой единичке из узора Лены соответствует включенное окно в доме Саши, потому что иначе Саше надо просить включить свет, но по условию задачи он этого не умеет делать. Сложность решения: $N \times M \times H \times W$.

Не забываем случай, когда узор состоит из всех нулей. В этом случае у Саши есть только один выход: попросить всех соседей выключить свет. Значит, ответ в этом случае: 1.

## Пример программы-решения

Ниже представлена вспомогательная программа на языке Python3

```
h_w, w_w = map(int, input().split())
windows = [list(map(int, input().split())) for i in range(h_w)]

h_p, w_p = map(int, input().split())
pattern = [list(map(int, input().split())) for i in range(h_p)]

if not any(map(any, pattern)):
    print(1)
```

```
else:
    ans = 0
    for y1 in range(h_w - h_p + 1):
        for x1 in range(w_w - w_p + 1):
            this_position_ok = True

            for y2 in range(h_p):
                for x2 in range(w_p):
                    if pattern[y2][x2] == 1 and windows[y1 + y2][x1 + x2] == 0:
                        this_position_ok = False

            ans += this_position_ok
    print(ans)
```

## *Задача 4.3.2. Собери свой учебный план-1 (10 баллов)*

Мальчик Слава очень любит учиться. В этом семестре ему предстоит изучить $n$ предметов, по каждому из которых существует несколько вариантов программы любой сложности. В институте, в котором учится Слава, сложность программы по предмету измеряется в зачётных единицах — это любое натуральное число, причём чем оно больше, тем сложнее считается программа.

Добрый замдекана факультета, на котором учится Слава, не разрешает студентам переусложнять свой учебный план на семестр, поэтому он разрешает студентам выбирать программы по предметам, только если их суммарная сложность равна зачётным единицам.

Как истинный комбинатор, Слава сразу увидел в сложившейся ситуации задачу — сколькими способами он может собрать свой учебный план на семестр, чтобы суммарная сложность программ по предметам была в точности равна $m$? Для каждого предмета в учебном плане Слава может выбирать любую сложность от 1 до $m$.

Поскольку ответ может быть слишком большой, Слава хочет знать остаток от деления его на $10^9 + 7$.

### Формат входных данных

В единственной строке входных данных записаны 2 натуральных числа $n (1 \leqslant n \leqslant 5)$ — число предметов в семестре и $m (1 \leqslant m \leqslant 10)$ — суммарная сложность программ, которую необходимо набрать Славе в семестре $(n \leqslant m)$.

### Формат выходных данных

В выходной файл выведите одно целое число — количество способов собрать учебный план по модулю $10^9 + 7$.

### *Пояснения к ответу*

В первом тесте из условий Слава может набрать свой учебный план 3 способами:

$$1 + 3$$

$$2+2$$

$$3+1$$

Обращаем ваше внимание на то, что порядок, в котором идут слагаемые в учебном плане, важен.

*Пример №1*

| Стандартный ввод |
|---|
| 2 4 |

| Стандартный вывод |
|---|
| 3 |

*Пример №2*

| Стандартный ввод |
|---|
| 3 6 |

| Стандартный вывод |
|---|
| 10 |

### *Решение*

В этой задаче всего $5 \times 10 = 50$ различных случаев. Решение — полный перебор или ручной подсчёт.

**Пример программы-решения**

Ниже представлена вспомогательная программа на языке Python3

```
from math import factorial


def binomial(n, m):
    return factorial(n) // factorial(m) // factorial(n - m)


n, m = map(int, input().split())
print(binomial(m - 1, n - 1) % (10 ** 9 + 7))
```

## *Задача 4.3.3. Собери свой учебный план-2 (15 баллов)*

Мальчик Слава очень любит учиться. В этом семестре ему предстоит изучить $n$ предметов, по каждому из которых существует несколько вариантов программы любой сложности. В институте, в котором учится Слава, сложность программы по предмету измеряется в зачётных единицах — это любое натуральное число, причём чем оно больше, тем сложнее считается программа.

Добрый замдекана факультета, на котором учится Слава, не разрешает студентам переусложнять свой учебный план на семестр, поэтому он разрешает студентам выбирать программы по предметам, только если их суммарная сложность равна зачётным единицам.

Как истинный комбинатор, Слава сразу увидел в сложившейся ситуации задачу — сколькими способами он может собрать свой учебный план на семестр, чтобы суммарная сложность программ по предметам была в точности равна $m$? Для каждого предмета в учебном плане Слава может выбирать любую сложность от 1 до $m$.

Поскольку ответ может быть слишком большой, Слава хочет знать остаток от деления его на $10^9 + 7$.

### Формат входных данных

В единственной строке входных данных записаны 2 натуральных числа $n, m (1 \leqslant n, m \leqslant 10^5)$ — число предметов в семестре и суммарная сложность программ, которую необходимо набрать Славе в семестре соответственно ($n \leqslant m$).

### Формат выходных данных

В выходной файл выведите одно целое число — количество способов собрать учебный план по модулю $10^9 + 7$.

#### ***Пояснения к ответу***

В первом тесте из условий Слава может набрать свой учебный план 3 способами:

$$1 + 3$$

$$2 + 2$$

$$3 + 1$$

Обращаем ваше внимание на то, что порядок, в котором идут слагаемые в учебном плане, важен.

*Пример №1*

| **Стандартный ввод** |
|---|
| `2 4` |

| **Стандартный вывод** |
|---|
| `3` |

*Пример №2*

| **Стандартный ввод** |
|---|
| `3 6` |

| **Стандартный вывод** |
|---|
| `10` |

### *Решение*

Даны два натуральных числа $n$ и $m$. Надо найти число решений в натуральных числах уравнения:

$$x_1 + x_2 + \cdots + x_n = m.$$

Выпишем в ряд $m$ единиц $\underbrace{111\ldots1}_{m}$.

Заметим, существует биекция из множества решений нашего уравнения в множество всевозможных способов поставить $n-1$ перегородку между этими единицами.

$$\underbrace{\underbrace{111\ldots1}_{x_1}|\underbrace{111\ldots1}_{x_2}|\ldots|\underbrace{111\ldots1}_{x_n}}_{m}$$

Способов так расставить перегородки, очевидно $C_{m-1}^{n-1}$. Осталось научиться считать это число сочетаний.

Можно посчитать на Python, не забыв взять ответ по модулю $10^9+7$. Если вы попробуете написать аналогичное решение на C++, вы неизбежно столкнетесь с проблемой переполнения на больших тестах. Значит, надо брать по модулю на промежуточных шагах. $a/b = a \cdot b^{-1} = a \cdot b^{MOD-2}$.

Сложность алгоритма: $O((m-n)(n \cdot \log_2((10^9+7)-2)))$.

#### Пример программы-решения

Ниже представлена вспомогательная программа на языке Python3

```python
from math import factorial


def binomial(n, m):
    return factorial(n) // factorial(m) // factorial(n - m)


n, m = map(int, input().split())
print(binomial(m - 1, n - 1) % (10 ** 9 + 7))
```

## *Задача 4.3.4. Расписание переговорной-1 (10 баллов)*

Переговорная комната "Яанровогереп"в компании "Скедня"каждый день открыта в течение $h$ часов. На завтра $n$ команд дали заявки на проведение в Яанровогерепе встреч. Для команды с номером известна величина $t_i$ — целое количество минут, которое им необходимо для решения всех вопросов. В один момент времени две команды не могут быть в переговорной, заходят и выходят они мгновенно. Необходимо узнать, какое максимальное количество команд могут посетить переговорную в этот день.

#### Формат входных данных

Первая строка содержит два целых числа $h, n (1 \leqslant h, n \leqslant 10)$ — время работы переговорки (в часах) и количество команд, желающих занять переговорку.

Вторая строка содержит $n$ целых чисел $t_i \leqslant (1 \leqslant t_i \leqslant 24 \cdot 60)$ — времена, на которые команды хотят занять переговорку (в минутах).

### Формат выходных данных

Выведите максимальное число команд, которые смогут посетить переговорную в этот день

### ***Пояснения к ответу***

В первом тесте переговорная будет свободна $2 \cdot 60 = 120$ минут, что является верхней границей суммарного времени групп в комнате. Рассматривая все возможные случаи, получаем, что если мы запустим первую группу, то больше четырех групп мы запустить не сможем. Тогда рассматриваем только группы со 2-й по 6-ю, суммарное время которых равно 120, то есть мы сможем их запустить, получив пять групп, что и будет ответом.

Во втором тесте единственная группа хочет занять комнату больше чем на то время, которое она будет открыта, поэтому никто не сможет зайти в нее.

В третьем тесте суммарное время, которое нужно всем группам равно 4 минутам, то есть мы сможем запустить все 4 группы.

*Пример №1*

| **Стандартный ввод** |
| --- |
| `2 6`<br>`70 20 30 40 10 20` |
| **Стандартный вывод** |
| `5` |

*Пример №2*

| **Стандартный ввод** |
| --- |
| `1 1`<br>`61` |
| **Стандартный вывод** |
| `0` |

*Пример №3*

| **Стандартный ввод** |
| --- |
| `3 4`<br>`1 1 1 1` |
| **Стандартный вывод** |
| `4` |

***Решение***

Переберем все возможные подмножества наших временных промежутков, для каждого проверим, удовлетворяет ли оно условию. Ответом будет максимальное количество промежутков в таком подмножестве.

Сложность алгоритма: $O(2^n)$.

### Пример программы-решения

Ниже представлена вспомогательная программа на языке Python3

```
h, n = map(int, input().split())
queries = list(map(int, input().split()))
total_time = h * 60

queries.sort()

cur_sum = 0
ans = 0
for q in queries:
    cur_sum += q
    ans += 1

    if cur_sum > total_time:
        ans -= 1
        break

print(ans)
```

## *Задача 4.3.5. Расписание переговорной-2 (20 баллов)*

Переговорная комната "Яанровогереп"в компании "Скедня"каждый день открыта в течение $h$ часов. На завтра $n$ команд дали заявки на проведение в Яанровогерепе встреч. Для команды с номером известна величина $t_i$ — целое количество минут, которое им необходимо для решения всех вопросов. В один момент времени две команды не могут быть в переговорной, заходят и выходят они мгновенно. Необходимо узнать, какое максимальное количество команд могут посетить переговорную в этот день.

### Формат входных данных

Первая строка содержит два целых числа $h (1 \leqslant h \leqslant 24), n (1 \leqslant n \leqslant 10^5)$ — время работы переговорки (в часах) и количество команд, желающих занять переговорку.

Вторая строка содержит $n$ целых чисел $t_i \leqslant (1 \leqslant t_i \leqslant 24 \cdot 60)$ — времена, на которые команды хотят занять переговорку (в минутах).

### Формат выходных данных

Выведите максимальное число команд, которые смогут посетить переговорную в этот день

### *Пояснения к ответу*

В первом тесте переговорная будет свободна $2 \cdot 60 = 120$ минут, что является верхней границей суммарного времени групп в комнате. Рассматривая все возможные случаи, получаем, что если мы запустим первую группу, то больше четырех групп мы запустить не сможем. Тогда рассматриваем только группы со 2-й по 6-ю, суммарное время которых равно 120, то есть мы сможем их запустить, получив пять групп, что и будет ответом.

Во втором тесте единственная группа хочет занять комнату больше чем на то время, которое она будет открыта, поэтому никто не сможет зайти в нее.

В третьем тесте суммарное время, которое нужно всем группам равно 4 минутам, то есть мы сможем запустить все 4 группы.

*Пример №1*

| Стандартный ввод |
|---|
| 2 6<br>70 20 30 40 10 20 |

| Стандартный вывод |
|---|
| 5 |

*Пример №2*

| Стандартный ввод |
|---|
| 1 1<br>61 |

| Стандартный вывод |
|---|
| 0 |

*Пример №3*

| Стандартный ввод |
|---|
| 3 4<br>1 1 1 1 |

| Стандартный вывод |
|---|
| 4 |

### *Решение*

Сортируем времена, необходимые группам, затем набираем временные промежутки, пока их сумма не превосходит итогового времени работы переговорной. Сложность решения $O(n \log n) + O(n) = O(n \log n)$.

## Пример программы-решения

Ниже представлена вспомогательная программа на языке Python3

```python
h, n = map(int, input().split())
queries = list(map(int, input().split()))
total_time = h * 60

queries.sort()

cur_sum = 0
ans = 0
for q in queries:
    cur_sum += q
    ans += 1

    if cur_sum > total_time:
        ans -= 1
        break

print(ans)
```

## *Задача 4.3.6. Списывание-1 (15 баллов)*

Приближаются школьные экзамены. В одной из школ их проводят так.

Детей рассаживают в аудиторию, в которой $a$ рядов, в каждом из которых $b$ одноместных парт. Ряды нумеруются один за другим, как и парты в одном ряду (ряды имеют номера от 1 до $a_1$, парты — от 1 до $b$ в каждому ряду).

Группа школьников решила попробовать списать на этом важном событии. Они решили, что будут передавать правильные ответы между собой, перекидываясь бумажками. Так как учителя не дремлют, то если школьник, сидящий в ряду $a_1$ за партой с номером $b_1$, кинет бумажку школьнику, сидящему в ряду $a_2$ за партой с номером $b_2$, то если $|a_1 - a_2| + |b_1 - b_2| > d$, то их заметят, и тогда план провалится. Так как это не первый экзамен в их жизни, многие пары уже были пойманы на списывании. За такими парами будет особый контроль, так что они не смогут перекидываться между собой. Для простоты вам будет дан список пар, которые еще не были замечены за списыванием друг у друга. Списывание удастся, если каждый сможет получить ответ от любого другого члена группы (возможно передавая через кого-либо). Смогут ли все школьники из группы всё-таки списать на экзамене?

### Формат входных данных

Первая строка содержит три целых числа $a, b, d (1 \leqslant a, b \leqslant 1000, a \cdot b \leqslant 10^4$, $0 \leqslant d \leqslant 10^9)$.

Вторая строка содержит два целых числа $n (1 \leqslant n \leqslant 10^4)$ — количество объединившихся для списывания человек, и $m \left(1 \leqslant m \leqslant \min\left(\frac{n(n-1)}{2}, 10^6\right)\right)$ — количество пар, люди в которых еще не списывали друг у друга (которые могут перекидывать бумажки).

Следующие $n$ строк содержат по два целых числа: в $i$-й строке указаны $a_i, b_i (1 \leqslant a_i \leqslant a, 1 \leqslant b_i \leqslant b)$, соответствующие $i$-му школьнику из списывающей группы. Гарантируется, что каждая пара $(a_i, b_i)$ встречается не более 1 раза.

Дальнейшие $m$ строк описывают пары, за которыми не будут повышенного надзора: в $i$-й строке содержатся два числа — $u_i, v_i (1 \leqslant u_i, v_i \leqslant n, u_i \neq v_i)$ — номера школьников из списка выше. Гарантируется, что каждая пара школьников указывается во входных данных не более 1 раза.

### Формат выходных данных

Выведите "YES"без кавычек, если списывание удастся, иначе выведите "NO".

#### *Пояснения к ответу*

В первом тесте каждый может перекидываться с каждым, но второй и третий находятся слишком далеко. Так как они могут передать ответы друг другу через второго, то каждый может передать каждому, ответ YES.

Во втором тесте первый находится слишком далеко от всех остальных, поэтому ответ NO.

*Пример №1*

**Стандартный ввод**
```
3 3 2
3 3
1 1
1 3
3 1
1 2
2 3
1 3
```
**Стандартный вывод**
```
YES
```

*Пример №2*

**Стандартный ввод**
```
3 4 2
3 3
1 1
1 4
3 4
1 2
2 3
1 3
```
**Стандартный вывод**
```
NO
```

#### *Решение*

Формализуем задачу: представим ее в виде графа на $n$ вершинах, где вершины соответствуют школьникам, а длина ребра - Манхэттенское расстояние между ними.

Для двух точек на плоскости считается как $d(x_1, y_1, x_2, y_2) = |x_2 - x_1| + |y_2 - y_1|$

Проведем все возможные ребра в нашем графе, запустим от какой-либо вершины поиск в глубину. Если мы сможем посетить все вершины, проходя только по разрешенным ребрам (которые указаны в тестах и их длина не превосходит d), то ответ YES, иначе NO.

### Пример программы-решения

Ниже представлена вспомогательная программа на языке Python3

```python
def get_p(v, p):
    if v == p[v]:
        return v
    p[v] = get_p(p[v], p)
    return p[v]


def union(a, b, p, size):
    a = get_p(a, p)
    b = get_p(b, p)

    if a != b:
        if size[a] > size[b]:
            p[b] = a
            size[a] += size[b]
        else:
            p[a] = b
            size[b] += size[a]


def dist(p1, p2):
    return abs(p1[0] - p2[0]) + abs(p1[1] - p2[1])


a, b, d = map(int, input().split())
n, m = map(int, input().split())

positions = [tuple(map(int, input().split())) for i in range(n)]

size = [1 for i in range(n)]
p = [i for i in range(n)]

for i in range(m):
    n1, n2 = map(int, input().split())
    n1 -= 1
    n2 -= 1

    if dist(positions[n1], positions[n2]) <= d:
        union(n1, n2, p, size)


common_p = get_p(0, p)
all_have_common_p = True
for i in range(1, n):
    if get_p(i, p) != common_p:
        all_have_common_p = False
        break

if all_have_common_p:
    print("YES")
```

```
else:
    print("NO")
```

### *Задача 4.3.7. Списывание-2 (20 баллов)*

Приближаются школьные экзамены. В одной из школ их проводят так.

Детей рассаживают в аудиторию, в которой $a$ рядов, в каждом из которых $b$ одноместных парт. Ряды нумеруются один за другим, как и парты в одном ряду (ряды имеют номера от 1 до $a_1$, парты — от 1 до $b$ в каждому ряду).

Группа школьников решила попробовать списать на этом важном событии. Они решили, что будут передавать правильные ответы между собой, перекидываясь бумажками. Так как учителя не дремлют, то если школьник, сидящий в ряду $a_1$ за партой с номером $b_1$, кинет бумажку школьнику, сидящему в ряду $a_2$ за партой с номером $b_2$, то если $|a_1 - a_2| + |b_1 - b_2| > d$, то их заметят, и тогда план провалится. Так как это не первый экзамен в их жизни, многие пары уже были пойманы на списывании. За такими парами будет особый контроль, так что они не смогут перекидываться между собой. Для простоты вам будет дан список пар, которые еще не были замечены за списыванием друг у друга. Списывание удастся, если каждый сможет получить ответ от любого другого члена группы (возможно передавая через кого-либо). Смогут ли все школьники из группы всё-таки списать на экзамене?

#### Формат входных данных

Первая строка содержит три целых числа $a, b, d (1 \leqslant a, b \leqslant 1000, a \cdot b \leqslant 10^4$, $0 \leqslant d \leqslant 10^9)$.

Вторая строка содержит два целых числа $n (1 \leqslant n \leqslant 10^5)$ — количество объединившихся для списывания человек, и $m \left(1 \leqslant m \ \leqslant \min\left(\frac{n(n-1)}{2}, 10^6\right)\right)$ — количество пар, люди в которых еще не списывали друг у друга (которые могут перекидывать бумажки).

Следующие $n$ строк содержат по два целых числа: в $i$-й строке указаны $a_i, b_i (1 \leqslant a_i \leqslant a, 1 \leqslant b_i \leqslant b)$, соответствующие $i$-му школьнику из списывающей группы. Гарантируется, что каждая пара $(a_i, b_i)$ встречается не более 1 раза.

Дальнейшие $m$ строк описывают пары, за которыми не будут повышенного надзора: в $i$-й строке содержатся два числа — $u_i, v_i (1 \leqslant u_i, v_i \leqslant n, u_i \neq v_i)$ — номера школьников из списка выше. Гарантируется, что каждая пара школьников указывается во входных данных не более 1 раза.

#### Формат выходных данных

Выведите "YES"без кавычек, если списывание удастся, иначе выведите "NO".

#### *Пояснения к ответу*

В первом тесте каждый может перекидываться с каждым, но второй и третий находятся слишком далеко. Так как они могут передать ответы друг другу через второго, то каждый может передать каждому, ответ YES.

Во втором тесте первый находится слишком далеко от всех остальных, поэтому ответ NO.

*Пример №1*

| **Стандартный ввод** |
|---|
| 3 3 2<br>3 3<br>1 1<br>1 3<br>3 1<br>1 2<br>2 3<br>1 3 |
| **Стандартный вывод** |
| YES |

*Пример №2*

| **Стандартный ввод** |
|---|
| 3 4 2<br>3 3<br>1 1<br>1 4<br>3 4<br>1 2<br>2 3<br>1 3 |
| **Стандартный вывод** |
| NO |

***Решение***

Формализуем задачу: представим ее в виде графа на $n$ вершинах, где вершины соответствуют школьникам, а длина ребра - Манхэттенское расстояние между ними.

Для двух точек на плоскости считается как $d(x_1, y_1, x_2, y_2) = |x_2 - x_1| + |y_2 - y_1|$

Будем хранить граф в виде списков смежности, добавляя только те ребра, которые нам указали в условии, и длина которых меньше d. Тогда если граф связный, то ответ YES, иначе NO. Критерием связности будет достижимость всех вершин из какой-либо одной, что проверяется поиском в глубину.

**Пример программы-решения**

Ниже представлена вспомогательная программа на языке Python3

```
def get_p(v, p):
    if v == p[v]:
        return v
    p[v] = get_p(p[v], p)
```

```python
    return p[v]


def union(a, b, p, size):
    a = get_p(a, p)
    b = get_p(b, p)

    if a != b:
        if size[a] > size[b]:
            p[b] = a
            size[a] += size[b]
        else:
            p[a] = b
            size[b] += size[a]


def dist(p1, p2):
    return abs(p1[0] - p2[0]) + abs(p1[1] - p2[1])


a, b, d = map(int, input().split())
n, m = map(int, input().split())

positions = [tuple(map(int, input().split())) for i in range(n)]

size = [1 for i in range(n)]
p = [i for i in range(n)]

for i in range(m):
    n1, n2 = map(int, input().split())
    n1 -= 1
    n2 -= 1

    if dist(positions[n1], positions[n2]) <= d:
        union(n1, n2, p, size)


common_p = get_p(0, p)
all_have_common_p = True
for i in range(1, n):
    if get_p(i, p) != common_p:
        all_have_common_p = False
        break

if all_have_common_p:
    print("YES")
else:
    print("NO")
```