Заключительный этап

Индивидуальный предметный тур

Математика. 8-9 класс

Задача III.1.1.1. (10 баллов)

Дом аттракционов виртуальной реальности предлагает два типа виртуальных пространств: VR-пространства со свободным перемещением игроков для тренировки ловкости и реакции, а также для путешествий по странам и музеям. На выбор предлагается 9 симуляторов первого типа и 6 — второго. Однако Дом имеет одно ограничение: под один аттракцион оборудована только одна комната и может принимать участие только один посетитель. Какова вероятность того, что пятеро одновременно пришедших в первый раз посетителей остановят свой выбор на трех VR-пространствах тренировки ловкости и реакции и двух аттракционах для путешествий? Ответ округлить до тысячных.

Решение

Всего аттракцион предлагает $9 + 6 = 15$ VR-пространств. Общее количество вариантов выбора 5 пространств из 15 равно

$$n = C_{15}^5 = \frac{15!}{5! \cdot 10!} = 3 \cdot 7 \cdot 11 \cdot 13.$$

Количество вариантов, когда возникает событие, благоприятствующее условию задачи, равно

$$m = C_9^3 \cdot C_6^2 = \frac{9!}{3! \cdot 6!} \cdot \frac{6!}{2! \cdot 4!} = 4 \cdot 5 \cdot 7 \cdot 9.$$

Тогда вероятность будет равна:

$$p = \frac{m}{n} = \frac{4 \cdot 5 \cdot 7 \cdot 9}{3 \cdot 7 \cdot 11 \cdot 13} = \frac{4 \cdot 5 \cdot 9}{3 \cdot 11 \cdot 13} = 0,420.$$

Ответ: $0,420$.

Задача III.1.1.2. (20 баллов)

Вычислить сумму корней уравнения $3(3x^2 - 5x - 5)^2 = 15x^2 - 24x - 20$. Ответ округлить до тысячных

Решение

Преобразуем уравнение к виду

$3(3x^2-5x-5)^2 = 5(3x^2-5x-5)+5+x <=>$

$3(3x^2-5x-5)^2 - 5(3x^2-5x-5) - 5 = x.$

Далее введем новую переменную $3x^2-5x-5=y$ и будем решать систему уравнений:

$$\begin{cases} 3x^2-5x-5=y, \\ 3y^2-5y-5=x. \end{cases}$$

Далее из верхнего уравнения отнимем нижнее, в результате чего получится выражение:

$(3x^2-3y^2)-5(x-y)+(x-y)=0,$

$(x-y)[3(x+y)-5+1]=0,$

$$\left[\begin{array}{l} x-y=0, \\ 3x+3y=4, \end{array}\right. => \left[\begin{array}{l} y=x, \\ y=\frac{4}{3}-x. \end{array}\right.$$

Подставляя каждое из этих равенств в верхнее уравнение исходной системы, получим два квадратных уравнения, откуда сможем найти корни данного уравнения:

$$\left[\begin{array}{l} 3x^2-5x-5=x, \\ 3x^2-5x-5=\frac{4}{3}-x, \end{array}\right. => \left[\begin{array}{l} x_{1,2}=1\pm\frac{2\sqrt{6}}{3}, \\ x_{3,4}=\frac{2}{3}\pm\frac{\sqrt{23}}{3}. \end{array}\right.$$

Отсюда видно, что сумма корней будет равна $(1+\frac{2\sqrt{6}}{3})+(1-\frac{2\sqrt{6}}{3})+(\frac{2}{3}+\frac{\sqrt{23}}{3})+(\frac{2}{3}-\frac{\sqrt{23}}{3})=\frac{10}{3}=3,333.$

Ответ: $3,333$.

Задача III.1.1.3. (20 баллов)

Копания студентов разработали игру — аналог китайских шашек, отличающуюся только формой игрового поля. Оно представляет собой шестиугольник, вершинами которого являются точки деления каждой стороны произвольного треугольника на три части. Найти площадь игрового поля, построенного на треугольнике площадью 100 см^2, если каждая сторона этого треугольника поделена в отношении 3:4:3. Ответ записать в см^2.

Решение

Пусть каждая из сторон делится в отношении $x:y:x$. Так как каждый из отсеченный треугольников подобен ΔABC (из-за пропорциональности двух сторон, содержащих общий угол, см. чертеж), то $\frac{S_1}{S}=\frac{S_2}{S}=\frac{S_3}{S}=\frac{x^2}{(2x+y)^2}$. Следовательно, площадь игровой поверхности равна

$S-3S\cdot\frac{x^2}{(2x+y)^2}=S(1-3\frac{x^2}{(2x+y)^2})=100(1-3\frac{3^2}{(10)^2})=73.$

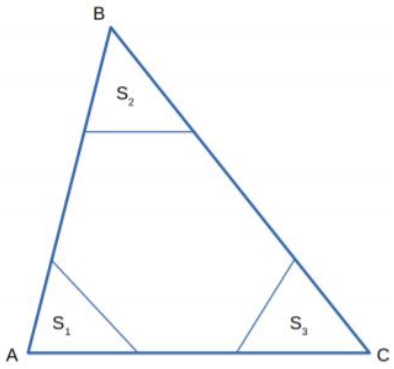


Рис. III.1.1: Чертеж к задаче

Ответ: 73.

Задача III.1.1.4. (25 баллов)

Игровое поле в парке виртуальной реальности представляет собой прямоугольный треугольник, который сверху сканируется тепловыми датчиками для определения положения игрока. Пользователь находится на пересечении медианы и биссектрисы, опущенных на разные катеты. При внезапном скачке напряжения в системе осталась запись о планируемой траектории пользователя, а также пройденном и оставшемся пути. Какова площадь, которую необходимо заново просканировать датчиками, если траекторией являлась упомянутая выше биссектриса, пройденное расстояние от вершины треугольника по этой биссектрисе равняется 36 метрам, а оставшееся — 20 метров?

Решение

Пусть точка — середина биссектрисы CB_1 (см. чертеж).

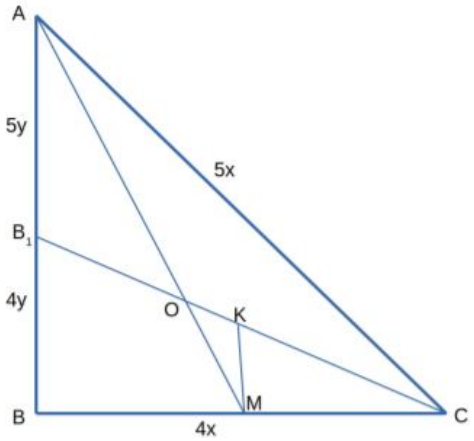


Рис. III.1.2: Чертеж к задаче

Тогда, очевидно, треугольники OKM и AOD являются подобными. Следовательно, $\frac{AD}{KM} = \frac{OD}{OK} = \frac{20}{8} = \frac{5}{2}$.

С другой стороны (т.к. KM — средняя линия треугольника BCD) $BD = 2KM$, откуда $\frac{AC}{BC} = \frac{AD}{BD} = \frac{5}{4}$. В таком случае обозначим стороны следующим образом $AB = 5y + 4y = 9y$, $AC = 5x$, $BC = 4x$. Используя теорему Пифагора, найдем связь между x и y:

$81y^2 + 16x^2 = 25x^2 => y = \frac{1}{3}x$.

Из выражения для длины биссектрисы получим уравнение относительно переменной x:

$CD^2 = BC \cdot AC - BD \cdot AD$,

$56^2 = 5x \cdot 4x - \frac{5}{3}x \cdot \frac{4}{3}$,

$x^2 = \frac{56^2 \cdot 9}{8 \cdot 20}$,

$x = \frac{42}{\sqrt{10}}$.

Отсюда мы сможем найти площадь игрового поля:

$x = \frac{1}{2}AB \cdot BC = \frac{1}{2}3x \cdot 4x = 6x^2 = 1058,4$.

Ответ: $1058,4$.

Задача III.1.1.5. (25 баллов)

Руководством компьютерного клуба было принято решение заняться майнингом, для чего было куплено три майнинг-фермы. Однако из-за перебоев с электричеством при вычислении одного набора хэшей в клубе была возможность запустить только одну ферму. Сначала была запущена первая ферма на такой промежуток времени, который потребовался бы для совместного выполнения всего объема вычислений на второй и третьей майниг-фермах. Далее была запущена вторая ферма. Она проработала столько часов, сколько потребовалось бы времени для совместного выполнения всего объема вычислений на первой и третьей фермах. Оставшиеся вычисления производились на третьей ферме в течение такого промежутка времени, который потребовался бы для совместного выполнения всего объема вычислений на первой и второй фермах. Во сколько раз быстрее были бы произведены вычисления одного набора хешей, если бы все три фермы выполняли работу одновременно и вместе?

Решение

Пусть объем работы, затрачиваемой на вычисление одного хэша, равно единице. Тогда количество работы, выполняемой первой, второй и третьей фермами, равно $\frac{1}{x}$, $\frac{1}{y}$, $\frac{1}{z}$.

Если бы все три фермы работали совместно, то за 1 час можно было бы выполнить объем вычислений, равный $\frac{1}{x} + \frac{1}{y} + \frac{1}{z} = \frac{xy+yz+xz}{xyz}$, а на выполнение всего объема вычислений затрачивалось бы $\frac{xyz}{xy+yz+xz}$ часов.

Время, затрачиваемое двумя фермами, будет равно:

$\frac{1}{\frac{1}{x}+\frac{1}{y}} = \frac{xy}{x+y}$ часов , если будут работать первая и вторая фермы;

$\frac{1}{\frac{1}{y}+\frac{1}{z}} = \frac{yz}{y+z}$ часов , если будут работать вторая и третья фермы;

$\frac{1}{\frac{1}{x}+\frac{1}{z}} = \frac{xz}{x+z}$ часов , если будут работать первая и третья фермы.

Тогда суммарно при последовательной работе фермы потратят $\frac{xy}{x+y} + \frac{yz}{y+z} + \frac{xz}{x+z}$ часов. При последовательном выполнении работы имеет место равенство:

$\frac{xy}{z\cdot(x+y)} + \frac{yz}{x\cdot(y+z)} + \frac{xz}{y\cdot(x+z)} = 1,$

так как весь объем работы за это время будет выполнен, согласно условиям задачи. Вычислим отношение времен:

$\left(\frac{xy}{x+y} + \frac{yz}{y+z} + \frac{xz}{x+z}\right) : \frac{xyz}{xy+yz+xz} = \frac{xy+yz+xz}{z(x+y)} + \frac{xy+yz+xz}{x(z+y)} + \frac{xy+yz+xz}{y(x+z)} =$

$= \frac{xy}{z(x+y)} + 1 + \frac{yz}{x(z+y)} + 1 + \frac{xz}{y(x+z)} + 1 = 4.$

Ответ: 4.

Математика. 10-11 класс

Задача III.1.2.1. (15 баллов)

На шлеме, использующемся в аттракционе виртуальной реальности, расположен оптический датчик, работу которого дублируют еще два точно таких же по характеристикам и надежности (т. е. вероятности безотказной работы) датчика. При выходе из строя центрального датчика происходит мгновенное переключение на правый либо на левый датчик. Определить надежность каждого из датчиков, если надежность всей системы дублирующих друг друга датчиков равна $0,992$.

Решение

Вероятность того, что одновременно откажут все три датчика, равны $(1-p)^3$, следовательно, вероятность того, что система будет работать безотказно, равна $P = 1-(1-p)^3$. Из этого выражения можно выразить вероятность $p = 1 - \sqrt[3]{1-P} = 0,8$.

Ответ: $0,8$.

Задача III.1.2.2. (20 баллов)

Бесконтактный лазерный 3D-сканер представляет из себя осветительный контур в виде правильной четырехугольной пирамиды (для считывания цветов сканируемого объекта) и контур сканирования формы объекта в виде куба, вписанного в осветительный контур. Во сколько раз больший объем пространства освещается, нежели нужно для непосредственного сканирования, если боковое ребро пирамиды составляет с плоскостью основания угол 60^o? Ответ записать с точностью до сотой.

Решение

Пусть даны четырехугольная пирамида $PABCD$ и вписанный в эту пирамиду куб
$A_1B_1C_1D_1A_2B_2C_2D_2$, причем точки A_1, B_1, C_1, D_1 принадлежат боковым ребрам пирамиды, а точки A_2, B_2, C_2, D_2 — ее основанию. Высота пирамиды — $PO, (PO \perp$

ΔABC). Обозначим угол между боковым ребром и основанием пирамиды как $\angle PAO = \alpha$ (см. чертеж), а ребро куба обозначим за x. Тогда объем куба

$$V_{\text{куба}} = x^3,$$

$OD_2 = \frac{x\sqrt{2}}{2}$. Из треугольника ΔD_1D_2D находим $DD_2 = x\operatorname{ctg}\alpha$, откуда $OD = \frac{x\sqrt{2}}{2} + x\cot\alpha$, $PO = OD \cdot \operatorname{ctg}\alpha = \frac{x(\sqrt{2}+2\operatorname{ctg}\alpha)}{2} \cdot \operatorname{tg}\alpha$. Следовательно, объем пирамиды

$$V_{\text{пир}} = \frac{1}{3} \cdot DB^2 \cdot PO = \frac{2}{3}\frac{x^2(\sqrt{2}+2\operatorname{ctg}\alpha)^2 \cdot x(\sqrt{2}+2\operatorname{ctg}\alpha)\operatorname{tg}\alpha}{4 \cdot 2} = \frac{\sqrt{2}x^3(1+\sqrt{2}\operatorname{ctg}\alpha)^3}{6\operatorname{ctg}\alpha}.$$

Таким образом, отношение объемов будет равно

$$\frac{V_{\text{пир}}}{V_{\text{куба}}} = \frac{\sqrt{2}x^3(1+\sqrt{2}\operatorname{ctg}\alpha)^3}{x^3 \cdot 6\operatorname{ctg}\alpha} = \frac{(1+\sqrt{2}\operatorname{ctg}\alpha)^3}{3\sqrt{2}\operatorname{ctg}\alpha} = 2,45.$$

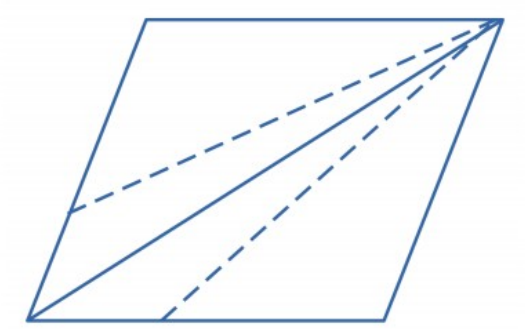
Ответ: 2,45.

Задача III.1.2.3. (15 баллов)

Таракан Янычар решил покончить со своим прошлым и замыслил побег от Артура Артуровича на воздушном змее. Рама воздушного змея заготовлена у соседского мальчишки и представляет собой ромб со стороной в 30 см, укрепленный еще и большей диагональю (см. сплошная линия на чертеже). Однако Янычар беспокоится, что из-за его веса змей переломится ровно по линии большой диагонали. Поэтому он решился на последнее преступление в своей недолгой жизни: украсть нитки и укрепить раму так, как показано на чертеже пунктирной линией. Какой длины должна быть нитка, если пунктирный линии разделят ромб на три равновеликие фигуры; косинус угла, за который крепятся и диагональ, и нитка, равен $\frac{1}{4}$, а на завязывание узлов на раме Янычару потребуется еще 4 см нитки?

Решение

Высоты треугольников CED и CFB, проведенные из вершины C (см. чертеж) имеют равные длины и $S_{\Delta CED} = S_{\Delta CFD}$ (по условию задачи); поэтому $DE = FB$, и, следовательно,
$CE = CF$ и $AE = AF$.



Диагональ в ромбе, делит четырехугольник $AECF$ на два равных треугольника. Отсюда можно увидеть, что $S_{\Delta ACF} = \frac{1}{2}S_{\Delta AECF}$. Так как по условию $S_{\Delta AECF} = S_{\Delta CFB}$, то $S_{\Delta ACF} = \frac{1}{2}S_{\Delta CFB}$, причем треугольники ACF CFB имеют общую высоту, проведенную из вершины C. Из этого вытекает, что $AF = \frac{1}{2}FB$, т. е. $FB = \frac{2}{3}a$; кроме того, $\cos B = \cos(180^o - C) = -\cos C = -\frac{1}{4}$. Из ΔCFB по теореме косинусов получим:

$$CF = \sqrt{a^2 + \frac{4a^2}{9} + 2a \cdot \frac{2a}{3}(-\frac{1}{4})} = \frac{4a}{3}.$$

Следовательно, $CE + CF = \frac{8a}{3} = \frac{8 \cdot 30}{3} = 80$ см.

Учтем дополнительную длину нити на завязывание узлов: $80 + 4 = 84$.

Ответ: 44.

Задача III.1.2.4. (20 баллов)

Найти скалярное произведение всех векторов, начала которых находятся в начале координат, а концы удовлетворяют условиям:

$$\begin{cases} 4x^2 - 2y^2 = \sqrt{2(2x+y)^2 - (2x+y)^4}, \\ y^4 + 2 \leqslant 4x(y^2 - 1). \end{cases}$$

Ответ записать с точностью до сотых.

Решение

В первом уравнении в правой части выражение $(2x + y)^2$ обозначим за z: $(2x + y)^2 = z$. Тогда подкоренное выражение правой части первого уравнения $2z - z^2$ будет представлять собой квадратичную функцию с наибольшим значением, равным 1.

Второе неравенство решим относительно переменной y: $y^2 \leqslant 2x + +\sqrt{4x^2 - 4x - 2}$. Далее данные в задаче условия можно переписать следующим образом:

$$\begin{cases} 4x^2 - 2y^2 \leqslant 1, \\ y^2 \leqslant 2x + \sqrt{4x^2 - 4x - 2}, \end{cases} => \begin{cases} 4x^2 \leqslant 2y^2 + 1, \\ y^2 \leqslant 2x + \sqrt{4x^2 - 4x - 2}, \end{cases} =>$$

$$\begin{cases} 4x^2 - 2y^2 \leqslant 1, \\ y^4 + 2 \leqslant 4x(y^2 - 1) \end{cases} => 4x^2 \leqslant 2y^2 + 1 \leqslant y^2 \leqslant 4x + 2\sqrt{4x^2 - 4x - 2} + 1$$

или

$4x^2 - 4x - 1 \leqslant 2\sqrt{4x^2 - 4x - 2}$, что эквивалентно равенству

$4x^2 - 4x - 2 - 2\sqrt{4x^2 - 4x - 2} + 1 \leqslant 0$,

$(\sqrt{x^2 - 4x - 2} - 1)^2 \leqslant 0$,

$\sqrt{x^2 - 4x - 2} = 1$,

$x = \left\{-\frac{1}{2}; \frac{3}{2}\right\}$.

Отсюда $y = \{0; -2\}$.

Таким образом, данным условиям удовлетворяют два вектора с концами в точках $\left(-\frac{1}{2}; 0\right)$ и $\left(\frac{3}{2}; -2\right)$, а их скалярное произведение равно $-0,75$.

Ответ: $-0,75$.

Задача III.1.2.5. (30 баллов)

При утверждении протокола проведения виртуальной конференции по созданию универсального тактильного костюма VR открытым научным сообществом "Virtu et Vérité"было приняты следующие решения:

1. проводить конференцию в пять этапов-сессий для одного или двух смежных часовых поясов;

2. закупить некий объем дискового пространства на сервере под выгрузку данных для совместной обработки, накопленных учеными;

3. перед первой сессией зарезервировать 280 Гб для программного обеспечения, которым смогут пользоваться участники конференции, после каждой сессии;

4. после каждой сессии организаторы сохраняют обработанные и заархивированные данные в том же облачном хранилище;

5. дисковое пространство во время проведения каждой сессии делится поровну и без остатка между всеми участниками заседания.

Определить, какой объем (в Гб) дискового пространства был закуплен организаторами, если в таблице ниже приведены сведения о номере сессии, количестве участников, между которыми распределялось и объеме данных, который сохранялся после каждой сессии:

Номер сессии	**Количество участников сессии**	**Объем архивов после сессии, Гб**
1	67	247
2	53	236
3	79	247
4	61	241
5	73	Оставшееся пространство

Решение

Поскольку резервирование указанных объемов серверного пространства приводит к тому, что оставшиеся гигабайты делятся нацело на количество участников конкретной сессии, можно сказать, что остаток деления суммы резервирований за предыдущие и текущую сессии на количество участников равен остатку деления искомого количества x гигабайт купленного пространства на количество участников. Т.е. $x = r_i \pmod{n_i}$:

$$\begin{cases} x = 12 \pmod{67}, \\ x = 50 \pmod{53}, \\ x = 52 \pmod{79}, \\ x = 34 \pmod{61}, \\ x = 53 \pmod{73}. \end{cases}$$

Далее можно воспользоваться китайской теоремой об остатках:

1. Максимальная оценка искомого числа составляет

$M = 67 \cdot 53 \cdot 79 \cdot 61 \cdot 73 = 1\ 249\ 195\ 637.$

2. Результат деления максимальной оценки искомого числа на количество участников $M_i = \frac{M}{n_i}$ равно:

$$\begin{cases} M_1 = 18\ 644\ 711, \\ M_2 = 23\ 569\ 729, \\ M_3 = 15\ 812\ 603, \\ M_4 = 20\ 112\ 269, \\ M_5 = 17\ 195\ 637. \end{cases}$$

3. Числа M_i^{-1}, обратные для M_i по модулю n_i (решение x уравнения $M_i \cdot x = 1 \pmod{n_i}$):

$$\begin{cases} M_1^{-1} = 41, \\ M_2^{-1} = 15, \\ M_3^{-1} = 32, \\ M_4^{-1} = 31, \\ M_5^{-1} = 14. \end{cases}$$

4. И таким образом, искомое число будет равно

$$x = \sum_{i=1}^{5} r_i \cdot M_i \cdot M_i^{-1} = 737\ 280 \text{ Гб}$$

Ответ: 737280.

Информатика. 8-11 класс

Задача III.1.3.1. Перепад глубины (100 баллов)

Автономный необитаемый подводный аппарат проплыл по заданной траектории, записав последовательность целых чисел a_i — глубину своего погружения в метрах на i-й секунде.

Известно, что за одну секунду глубина погружения не может измениться более чем на d метров в большую или меньшую сторону. Определение глубины не всегда работает точно, из-за чего это условие может быть нарушено в исходной последовательности. Необходимо очистить данные, удалив из них как можно меньше элементов.

Требуется написать программу, которая по данным a_i и d определит наименьшее возможное количество элементов, которые нужно удалить из последовательности a_i чтобы разность рядом стоящих элементов в оставшейся последовательности по модулю не превосходила d.

Формат входных данных

Входные данные содержат целые числа n и d — количество измерений и максимальный перепад глубины. Далее следует n целых чисел a_i — измерения глубины.

Формат выходных данных

Выходные данные должны содержать единственное целое число — наименьшее количество удаляемых элементов.

Ограничения

$1 \leqslant n \leqslant 20000,\ 0 \leqslant d \leqslant 1000,\ 0 \leqslant a_i \leqslant 10^9$.

Примеры

Пример №1

Стандартный ввод
3 4 10 15 14
Стандартный вывод
1

Решение

Задача решается методом динамического программирования.

Пусть dp_i — минимальное количество элементов, которые нужно удалить, чтобы элементы на префиксе с 1-го по i-й имели перепад не более d, и при этом элемент i остался в последовательности.

Тогда $dp_i = 0$ и $dp_i = \min\limits_{j=\overline{1,i-1}} dp_j + i_j - 1, i = \overline{2,n}$.

следует также учесть возможность удаления элементов в суффиксе последовательности, поэтому окончательный ответ равен $\min\limits_{i=\overline{1,n}} dp_i + n - i$

Пример программы-решения

Ниже представлено решение на языке C++.

```cpp
#include <algorithm>
#include <iostream>
#include <vector>

int main() {
    int n, d;
    std::cin >> n >> d;
    std::vector<int> a;
    a.resize(n);
    for (int i = 0; i < n; ++i) std::cin >> a[i];
    std::reverse(a.begin(), a.end());
    std::vector<int> dp;
    dp.resize(n);
    for (int i = 0; i < n; ++i) {
        dp[i] = i;
        for (int j = 0; j < i; ++j) {
            int v = dp[j] + i - j - 1;
            if (std::abs(a[i] - a[j]) <= d && v < dp[i])
                dp[i] = v;
        }
    }
    int m = dp[n - 1];
```

```
23      for (int i = 0; i < n - 1; ++i)
24          m = std::min(dp[i] + n - i - 1, m);
25      std::cout << m;
26      return 0;
27  }
28  }
```

Задача III.1.3.2. Отладка шарика (100 баллов)

Юный программист Вася решил разработать собственную платформу виртуальной реальности. Для начала он реализовал вывод пустой черной сцены с единственным красным шаром. Однако в алгоритме вывода шара Вася допустил ошибки, из-за которых шар не всегда выводился правильно.

Для отладки Вася сделал скриншоты своего приложения (шар на них должен превратиться в круг). Скриншот представлен прямоугольным массивом символов, в котором символ «#» представляет красный пиксель, а символ «.» — черный пиксель.

Если шар нарисован правильно, то красными будут только те пиксели, координаты которых
$(x; y)$ удовлетворяют условию $(x - x_0)^2 + (y - y_0)^2 \leqslant r^2$, где $x_0,\ y_0,\ r$ — неизвестные целые числа.

Требуется написать программу, которая по изображению выяснит, правильно ли на нем нарисован шар (спроецированный в круг), и если да, то определит значения $x_0,\ y_0,\ r$.

Формат входных данных

Первая строка входных данных содержит целые числа $X\ Y$ — ширину и высоту прямоугольника. Следующие Y строк содержат по X символов «#» и «.» каждая — описание изображения.

Формат выходных данных

Выходные данные должны содержать целые числа $1 x_0 y_0 r$, если изображение является правильным, и число 0 в противном случае. Координаты отсчитываются от верхнего левого угла.

Ограничения

$$3 \leqslant X, Y \leqslant 2000,\ r \geqslant 1.$$
$$1 \leqslant x_0 - r < x_0 + r \leqslant X,\ 1 \leqslant y_0 - r < y_0 + r \leqslant Y.$$

Примеры

Пример №1

Стандартный ввод
3 3 .#. ### .#.

Стандартный вывод
1 2 2 1

Решение

Поскольку правильное изображение содержит ровно один круг, то ограничивающий прямоугольник всех красных пикселей на изображении должен совпадать с ограничивающим прямоугольником круга.

Таким образом, следует найти ограничивающий прямоугольник (то есть найти минимальные и максимальные координаты красных пикселей по каждой оси), проверить, что он является квадратом с нечетной длиной стороны, найти потенциальный центр круга как центр ограничивающего квадрата. Затем следует проверить, что внутри квадрата красными являются те и только те пиксели, которые удовлетворяют условию:

$$(x - x_0)^2 + (y - y_0)^2 \leqslant r^2.$$

Пример программы-решения

Ниже представлено решение на языке C++.

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <cmath>

static int X, Y;
static std::vector<std::string> a;

static int sqr(int x) { return x*x; }
static int sqrt(int x) { return int(std::sqrt(x)); }

int best_circle_at(int xc, int yc) {
    int minr = 0, maxr = sqr(X);
    for (int y = 0; y < Y; ++y) {
        for (int x = 0; x < X; ++x) {
            int r2 = sqr(x - xc) + sqr(y - yc);
            if (a[y][x] == '#')
                minr = std::max(r2, minr);
            else
                maxr = std::min(r2 - 1, maxr);
        }
    }
    return maxr >= minr ? sqrt(maxr) : 0;
}

bool check_circle_at(int xc, int yc, int r) {
    for (int y = 0; y < Y; ++y) {
        for (int x = 0; x < X; ++x) {
```

```
30              int r2 = sqr(x - xc) + sqr(y - yc);
31              if ((a[y][x] == '#') != (r2 <= sqr(r)))
32                  return false;
33          }
34      }
35      return true;
36  }
37
38  int main() {
39      std::cin >> X >> Y >> std::ws;
40      a.resize(Y);
41      for (int y = 0; y < Y; ++y)
42          std::getline(std::cin, a[y]);
43      int bestr = 0, bestx, besty;
44      for (int y = 0; y < Y; ++y) {
45          for (int x = 0; x < X; ++x) {
46              int r = best_circle_at(x, y);
47              if (r > bestr) {
48                  bestr = r;
49                  bestx = x;
50                  besty = y;
51              }
52          }
53      }
54
55      if (bestr == 0 || bestr > X / 2 || !check_circle_at(bestx, besty, bestr)) {
56          std::cout << 0;
57      }
58      else {
59          std::cout << "1 " << bestx + 1 << " " << besty + 1 << " " << bestr;
60      }
61  }
62  }
```

Задача III.1.3.3. Морской бой (100 баллов)

Вася создает игру про корабли. В игре есть мины, которые имеют форму круга. Корабль имеет форму выпуклого многоугольника. Вася хочет придумать механику взаимодействия мины и корабля, для этого ему требуется вычислить площадь пересечения мины и корабля. Так как у Васи плохо с геометрией, он просит вас написать программу для вычисления площади пересечения корабля, заданного координатами вершин, и мины, заданной радиусом и координатами центра.

Формат входных данных

Первая строка входных данных содержит 3 целых числа x_c, y_c, r_c — координаты и радиус мины. Вторая строка содержит одно целое число N — количество вершин в многоугольнике, описывающем корабль. Следующие N строк содержат N пар чисел (x_i, y_i) (по одной паре в строке) — координаты вершин выпуклого многоугольника, описывающего корабль, в порядке обхода по часовой стрелке.

Формат выходных данных

Выходные должны содержать одно вещественное число — площадь пересечения мины и корабля с точностью не менее двух знаков после запятой.

Ограничения

$$-1000 \leqslant x_c, y_c, r_c, x_i, y_i \leqslant 1000$$
$$3 \leqslant N \leqslant 10^3$$

Описание подзадач и системы оценивания

Решения, работающие для $-10 \leqslant x_c, y_c, r_c, x_i, y_i \leqslant 10$, оцениваются из 50 баллов. Баллы выставляются за каждый успешно пройденный тест.

Примеры

Пример №1

Стандартный ввод
1 1 1 4 0 0 0 1 1 1 1 0
Стандартный вывод
0.785398163

Пример №2

Стандартный ввод
0 0 1 3 0 0 0 1 1 0
Стандартный вывод
0.5

Пример №3

Стандартный ввод
0 0 4 4 0 0 0 2 4 4 2 0

Стандартный вывод
7.07035

Пример программы-решения

Ниже представлено решение на языке C++.

```
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <algorithm>
#include <map>
#include <math.h>
#include <queue>
#include <set>
#include <iostream>
#include <iomanip>


using namespace std;

pair<double, double> points[100000];

double intersect_two_section(double x1, double x2, double xx1, double xx2)
{
        if (x2 < xx1 || x1 > xx2)
                return 0;
        if (x1 >= xx1 && x1 <= xx2 && x2 >= xx1 && x2 <= xx2)
                return x2 - x1;
        if (xx1 >= x1 && xx1 <= x2 && xx2 >= x1 && xx2 <= x2)
                return xx2 - xx1;
        if (x1 >= xx1 && x1 <= xx2)
                return xx2 - x1;
        else
                return x2 - xx1;
}

int main() {
#ifndef ONLINE_JUDGE
        freopen("input.txt", "r", stdin);
        freopen("output.txt", "w", stdout);
#endif
        ios_base::sync_with_stdio(0);
        cin.tie(0);
        int n;
        double x, y, r;
```

```
44          cin >> x >> y >> r;
45          cin >> n;
46          double min_x = LONG_MAX, max_x = -LONG_MAX;
47          for (int i = 0; i < n; i++)
48          {
49                  cin >> points[i].first >> points[i].second;
50                  points[i].first -= x; points[i].second -= y;
51                  min_x = min(min_x, points[i].first);
52                  max_x = max(max_x, points[i].first);
53          }
54          int pointer = 0;
55
56          while (points[(pointer + 1) % n].first >= points[pointer].first)
57                  pointer = (pointer + 1) % n;
58          while (points[(pointer + 1) % n].first <= points[pointer].first)
59                  pointer = (pointer + 1) % n;
60
61          int left1 = pointer;
62          int left2 = (pointer - 1);
63          if (left2 < 0)
64                  left2 = n - 1;
65
66          int right1 = pointer;
67          int right2 = (pointer + 1) % n;
68
69          double step = (max_x - min_x) / 20000000;
70          step = max(step, 0.000001);
71          bool st = false;
72          double last_sec = -1;
73          double ans = 0;
74          double eps = 0.000000001;
75          for (double pos = min_x; pos <= max_x; pos += step)
76          {
77             if (r * r >= pos * pos)
78             {
79                double p1 = -sqrt(r * r - pos * pos);
80                double p2 = sqrt(r * r - pos * pos);
81                while (pos > points[right2].first)
82                {
83                        right1 = right2;
84                        right2 = (right1 + 1) % n;
85                }
86                while (pos > points[left2].first)
87                {
88                        left1 = left2;
89                        left2 = left1 - 1;
90                        if (left2 < 0)
91                                left2 = n - 1;
92                }
93                double delta_y1 = points[right2].second - points[right1].second;
94                double delta_y2 = points[left2].second - points[left1].second;
95                double Y1 = delta_y1 * (pos - points[right1].first)
96                 / (points[right2].first - points[right1].first) + points[right1].second;
97                double Y2 = delta_y2 * (pos - points[left1].first)
98                / (points[left2].first - points[left1].first) + points[left1].second;
99                double new_sec = intersect_two_section(Y2, Y1, p1, p2);
100               if (st)
101               {
102                       ans += (new_sec + last_sec) * step / 2;
103               }
```

```
104                 st = true;
105                 last_sec = new_sec;
106             }
107         }
108         cout << setprecision(20) << fixed;
109         cout << ans;
110 }
```