Олимпиада школьников «Надежда энергетики»

Москва

Место проведения

ZZ94-84

шифр

← Не заполнять
Заполняется ответственным работником

Вариант № 73991

ФАМИЛИЯ Аршинов

ИМЯ Владислав

ОТЧЕСТВО Павлович

Дата рождения 20.10.2002

Класс: 9

Предмет Информатика

Этап: Заключительный

Работа выполнена на 2 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№5.

Делаем цикл, который работает, пока ~~в шифре~~ шифр не пустой. На каждой итерации проверяем начало шифра: если первые 6 букв разные, а 7-я совпадает с 3-й, то вот он „подъезд" — и после этого удаляем первую букву шифра.

(+)

№2.

Рассмотрим первый ряд карточек. Нам нужно его отсортировать по неубыванию. Для этого мы сначала ищем карточку (или карточки) с минимальным числом и кладем ее (их) в конец ряда. ~~И так,~~ Затем мы снова начинаем искать минимальное число, но уже не считая ту карточку (те карточки) ~~что уже в~~ которую (которые) мы положили в конец ряда. Повторяем это до тех пор, пока ряд не будет полностью отсортирован.

Что касается второго ряда, там такой же алгоритм, только вместо минимальных мы ищем максимальные числа.

+ 9

№4.

Во-первых, чисто математическое (Даже скорее логическое.) решение задачи. В последовательности есть позиции, на которых элементы не меняются. Это позиции, ~~с индексом~~ у индекса которых остаток от деления на (10) 1, 5, 6, 9 и 0, поскольку в любой степени эти цифры остаются последними. А что касается остальных: 2, 3, 4, 7, 8 — последняя цифра ~~этих~~ степеней этих цифр зависит от остатка от деления индекса на (4), например, 2^1 заканчивается на 2, 2^2 – на 4, 2^3 – на 8, 2^4 – на 6, а потом снова то же самое. ~~Всего на цифр. у нас~~ Следовательно, длина наименьшего периода последовательности является наименьшим общим кратным 10 и 4, то есть 20.

Во-вторых, алгоритмическое решение. Делаем цикл из 10! итераций, и при каждой пишем в список или массив результат — число в последовательности. А потом начинаем искать период: пробуем длину 1, 2, 3, 10 и т.д.

+

№3.

Если Серёжа n раз делил на b, то число в результате будет $\frac{a}{b^n}$. ~~Знамен~~ Чем больше n, тем больше знаменатель, тем меньше само число. В конце концов число становится ~~на~~ таким маленьким, что в ячейках памяти калькулятора остались только нули.

∓

№1.

Число yx может принимать значения от 10 до 99 включительно. Делаем цикл, и для каждого yx_i ~~ставим в~~ вычисляем A_i и проверяем на делимость второго на первое.

±

Олимпиада школьников «Надежда энергетики»

Москва

Место проведения

АИ 32-91

шифр

← **Не заполнять**

Заполняется ответственным работником

Вариант № 73101

ФАМИЛИЯ Волянский

ИМЯ Дмитрий

ОТЧЕСТВО Ильич

Дата рождения 10.12.2002

Класс: 10

Предмет ИНФОРМАТИКА

Этап: ЗАКЛЮЧИТЕЛЬНЫЙ

Работа выполнена на 3 **листах**

Дата выполнения работы: 17.02.19

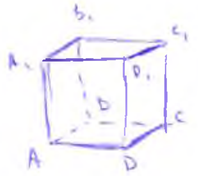
(число, месяц, год)

Подпись участника олимпиады:

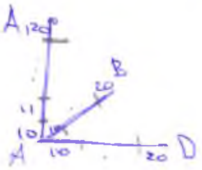
Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

№1.

Нарисуем куб со стороной 11:



Рассмотрим ребра AA1; AB и AD:



разобьем их на 11 равных отрезков каждое и пронумеруем эти части от 10 до 20 начиная с A =>

Если теперь мы разобьем наш кубик на более маленькие кубики плоскостями, || граням и проходящими через точки деления отрезков AA1; AB и AD на 11 частей =>

Куб разбился на 1331 более маленьких частей, причем у каждой части есть "координаты" по x (AA1), y (AD) и z (AB) => наши маленьких кубики "пробегают" все возможные тройки (x, y, z)

+

Тогда мы можем для каждого кубика проверить удовлетворяют ли его координаты уравнению $3x^2-y^2-7z=99$. Если да, то этот кубик мы запоминаем, то есть записываем тройку его координат $(x_i; y_i; z_i)$.

Т.к. кубиков конечное число то алгоритм закончится, а в конце него мы найдём все отвечающие комбинации.

№3.

Заметим, что такое могло случиться, если $|b|>1$. Тогда $\lim\limits_{n\to\infty}\frac{a}{b^n}=0$ => рано или поздно из-за действий серёжи в обычных вычислениях мы получим число сколь угодно близкое к нолю.

Из-за ограниченности экрана и памяти, калькулятор запоминает только несколько первых знаков с округлением, то есть если калькулятор может запомнить только m знаков с округлением, то когда число серёжи станет меньше, чем $\underbrace{0,0\ldots0}_{m\text{-нолей}}4$, то калькулятор выведет "0". А такой момент настанет, т.к. мы можем получить число сколь угодно близкое к нолю через конечное число действий.

2

±

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№5

1.Если мы хотим проверить, являются ли а и в искомыми, то в мы можем заменить на в', а на а', где в' и а' - остатки при делении а и в на 10. Так сделать можно, т.к $cd \underset{10}{\equiv} r_c r_d$, где r-остатки, а $c \underset{10}{\equiv} r_c$ => $f(x) = в \cdot (x^3 + 7x^2 + 3x + a) \underset{10}{\equiv} в'(x^3 + 7x^2 + 3x + a')$ => остаток не поменялся.

±

Тогда для чисел от 0 до 9 мы запомним своё число y_i (i - цифра), где y_i -остаток [illegible] $в'(i^3 + 7i^2 + 3i + a)$ при делении на 10. Если все значения y_i - различны, то расшифровать можно однозначно => а и в - искомые.

2. Для того, чтобы найти какие-то значения а и в мы можем ~~най~~ (из аналогичных первому пункту соображений) найти а' и в' (остатки), которое нам подходят для однозначной расшифровки. То есть для всех а от 0 до 9 мы подбираем в пару в от 0 до ~~8~~9 (всего 100 пар). И каждую пару проверяем по алгоритму из 1 пункта. тогда для каждой "хорошей" пары (а', в') мы можем построить бесконечное множество пар (а, в) где

$$\begin{cases} a = a' + 10k \\ в = в' + 10R \end{cases}$$

, где k и R $\in N \cup \{0\}$ (Но если $a' = 0$, то $k > 0$, если $в' = 0$, то $R > 0$)

Если пар (а', б') не нашлось, то числа (а, б) тоже не существует.

№4.

1 Создадим таблицу 2×N:

Теперь для каждого i от 1 до N в i столбце мы будем записывать сумму чисел в закрашенных клетках (в 1 строке) и максимальное значение (во 2 строке)

2. Приведём алгоритм для i-той строки таблицы N×N, который поможет найти необходимое значения: Пусть $S_0 = 0$, а m_0 равно первому значению в i строке (не считая пустые клетки). Если в закр клетках числа не нашлось, то в i столбце во 2 строке мы оставим пустое место.

3. Рассмотрим параметр k, который будет ~~про~~ пробегать целые значения от 1 до N-i, для $i \leq [\frac{N+1}{2}]$ и от 1 до i-1, для $i > [\frac{N+1}{2}]$ (тогда в i строке параметр k пробежит только номера всех закраш. клеток).

~~до~~ Для каждого такого k мы сделаем (если клетка не пустая):

1) $S_k = S_{k-1} + a_k$

2) ~~как~~ Если $a_k > m_{k-1}$ => $m_k = a_k$, иначе $m_k = m_{k-1}$

где a_k - число на k месте i-той строки.

Если клетка с номером k пуста, то:

1) $S_k = S_{k-1}$

2) $m_k = m_{k-1}$

±

4. Последние значения S_k и m_k (то есть при max для i строки k) мы вносим в i-той столбец таблицы 2×N.

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ AU 32-91

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

5. Если 2 строка таблицы 2×N пуста => в закр. части таблицы N×N чисел не было

Иначе, взяв аналогично параметр k (принимает значение от 1 до N) ~~находим SN и так для таблицы 2×N [illegible] сложив все~~ в 1 строке сложим все вk (вk - число на k-той позиции) (s0=0 *Если k-тая клетка пуста, то sk=sk-1, иначе sk=sk-1+вk) во 2 строке среди всех ck (ck - число на k-той позиции) найдем max (Если k-тая клетка пуста, то mk=mk-1, иначе, если ck>mk-1 => mk=ck, иначе mk=mk-1, где m0=первый непустой элемент в строки).

SN и mN - искомое значение.

№2

Т.к. сторона треугольника ограничена, столько, что можем найти конечное а, в и с - длины 1, 2 и 3 ряда соответственно

Если
$\begin{cases} a+b>c \\ a+c>b \\ b+c>a \end{cases}$ => неверные данные, т.к. это противоречит н-ву Δ.

1 Чтобы упорядочить ряд от наименьшего элемента к большему, ~~Воспользуемся методом пузырька. [illegible] для k от одного до а, [illegible] ek [illegible] на k-той карточке [illegible] пока [illegible] ek [illegible]~~ для начала найдем минимальный элемент, среди e1… ea (ei - элемент на i-той карточке) по след. алгоритму: m0=e1; mi=mi-1, если mi-1<ei, иначе mi=ei (для i=1 до а). Тогда найдем ek=ma. Тогда ma - значение минимального элемента. И поменяем его местами с первым элементом данной группы (e1)

2. Тогда мы получили, что минимальный элемент на первом месте => для e2…ea проводим аналогичное действие (ej:=ma <-> e2) => в конце-концов мы упорядочим ряд от наименьшего элемента к наибольшему (каждый элемент не превосходит предыдущий, среди нескольких е-мент, включая ~~[illegible]~~ вед предыдущий - текущую) => получили e1'…ea'

3. d2…dв - элементы второй строки. Будем делать аналогично с первой строке, только найденный минимум меняем местами с последним элементом => получили d2'…dв'

4. f2…fc-1 - элементы 3 строки. Для них проведём алгоритм, как для первой строки => получим f2'…fc-1' - упорядоченную последовательность

=> вторую строку мы тоже упорядочили => Тогда если f2'<dc' => 'На стыке со 2 рядом не оказался минимум!

если fc-1'>a1' => 'На стыке с 1 рядом не оказался максимум!

Иначе - все в хорошо.

Лист 3 из 3

Олимпиада школьников «Надежда энергетики»

	МОСКВА
№ группы	Место проведения

ДК 50-10

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Дукин

ИМЯ Никита

ОТЧЕСТВО Олегович

Дата рождения 23.09.2001

Класс: 11

Предмет информатика

Этап: заключительный

Работа выполнена на 7 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, общее количество листов, на которых выполнена работа и дату выполнения работы.

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ДК 50-10

N1

Дана последовательность чисел ~~пд~~ $C_1, C_2, C_3 \dots C_n$, в которой посл. цифр. $- n^2$. Давайте составим таблицу первых элементов.

Элемент n	1	2	3	4	5	6	7	8	9	10	11	12
Значение (посл. цифра от n^2)	1	4	7	6	5	6	3	4	9	0	1	6

Элемент значение

$1^1 \Rightarrow \textcircled{1}$

$2^2 = 2 \cdot 2 \Rightarrow \textcircled{4}$

$3^3 = 3 \cdot 3 \cdot 3 = 27 \Rightarrow \textcircled{7}$

$4^4 = \underbrace{4 \cdot 4}_{16} \cdot 4 \cdot 4 \Rightarrow \textcircled{6}$

16 / ...4 / ...6

$16 \cdot 4 \cdot ?$

$$\begin{array}{r} \times\, 16 \\ 4 \\ \hline 16 \\ 4 \\ \hline \dots 6 \end{array}$$

понимаем, что последняя цифра числа зависит только от произведения последних цифр ⇒ мы можем перемножать лишь крайние числа ненужный раз.

$5^5 = \underbrace{5 \cdot 5}_{25} \cdot 5 \cdot 5 \cdot 5 \Rightarrow \textcircled{5}$

25 / ...5 / ...5 / ...5

$6^6 = \underbrace{6 \cdot 6}_{36} \cdot 6 \cdot 6 \cdot 6 \cdot 6 \Rightarrow \textcircled{6}$

36 / ...6 / ...6 / ...6 / ...6

. . . .

$12^{12} = \underbrace{12 \cdot 12}_{\dots 4} \cdot 12 \cdot 12 \cdot 12 \cdot 12 \cdot 12 \cdot 12 \cdot 12 \cdot 12 \cdot 12 \cdot 12 \Rightarrow \textcircled{6}$

...4 / 8 / ...6 / ...2 / ...4 / ...8 / ...6 / ...2 / ...4 / 8 / ...6

Понимаем, что можно брать ~~от~~ последнюю цифру двузначных и многозначных чисел и перемножать n раз их.

Чтобы взять последнюю цифру, нужно рассмотреть число и взять остаток от деления на десять (10), т.е. $x \bmod 10$, где x - номер элемента в последовательности. И, перемножая x икс раз ~~получаем~~ и беря mod 10, получаем искомое значение.

Для нахождения периода посл-ти, не большего ~~н~~ 10! будем пробегаться по ~~ст~~ массиву последов-ти и добавлять в массив, где каждому эл-ту соответствует массив из двух элементов — само число и его индекс. Как найдем

Лист 1 из 1

№1 (продолжение)

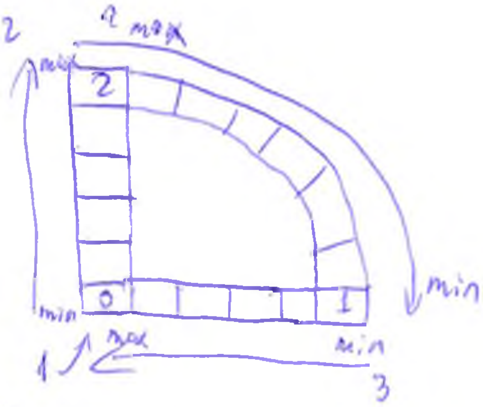
таким же, будем сравнивать ~~индексы~~ элем.-ты с индексами i и $i+n$, где i - индекс первого числа, а $i+n$ - второго, n - разница индексов, запомним сначала $c = i+n$. Если $i \geq c$, то найден мин. период - с i до $i+n$ (i - включ., $i+n$ - не вкл, т.к. в массиве эти индексы хранят один. эл-ты). Если вдруг есть расхождение в элементах, то добавляем новый элемент в наш ~~массив~~, двумерный массив и в следующий раз будем сравнивать с i и $i+n$ столько раз, сколько вхождений числа уже было, ведь может оказаться, что в периоде несколько одинаковых чисел. Цикл будет перебирать от 1 до 10!, т.е. $10 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$. включит. ведь в условии макс. период - 10!

Алг-м.:

```
1. Вводим массив A
2. ~~Проверяем~~:       | от i=1 эл. до последнего (длина массива):
(проверка)              |     если A[i] - не число, то выводим ошибку
                        |     $K = A[i] \bmod 10$
                        |     от J=1 до i (включ.):
                        |         $J = (J \bmod 10) \cdot (J \bmod 10)$
                        |     если $A[i] \neq J$:
                        |         выводим ошибку
3. B = [ ] (пустой массив)
   от i=1 до $10 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$:
       если A[i] есть в массиве, то
           от J=2 до конца массива двумерного массива с индексом вхождения A[i]:
               r = J-му элементу [J-ое вхождение в массив)
               пока $r = i+n$, то
                   если $r \geq i$:
                       выводим массив A от r до i+n
                       выходим из циклов!
                   иначе.
                       $r = r+1$
                       $n = n+1$.
4. Если ничего не выведено, то выводим, что длина периода > 10!
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№2



Обход указан на рисунке.
Если отсортировать, как требуется (см. рисунок), то возможны ~~два~~ три случая нарушения посл-ти: в положении 0 и в пол-ии 1, и в 2.
Нужно запомнить элементы 0 и 1 и 2.

Если эл-ты 0 и 1 после трёх сортировок не соответствуют начальным значениям, то выведем, что с другими рядами min или max невозможен. (*)

Для сортировки будем использовать эффективную цифровую сортировку: пусть у нас 11 массивов. B – массив с цифрами, которые нужно отсортировать и 10 массивов, где каждому будет соответствовать номер A_0 – 0, A_1 – 1 … A_9 – 9. Пробегаемся по массиву B. Смотрим mod 10 числа. Если B[i] mod 10 = 0, то добавляем в A_0, если 1, то в A_1 и т.д. И удаляем из массива B. Опять перегоняем все числа в массив B с отсортированными по последн. числу цифрами. ~~Иначе~~ Теперь пробегаемся снова по массиву B. И в очищенные массивы A_0..A_9 добавляем числа, отсортированные по десяткам, то есть B[i] mod 100 = 0, то добавляем в A_0 и так далее. И последовательно перегоняем числа обратно в очищенный массив B. Теперь в B числа, отсортированные по 2-ум посл. числам. Продолжаем делать так до мах. разр. макс. элемента и рассматриваем mod 1000, потом mod 10000 и так далее. Теперь в B числа отсортированные за $n \log_{10} n$!

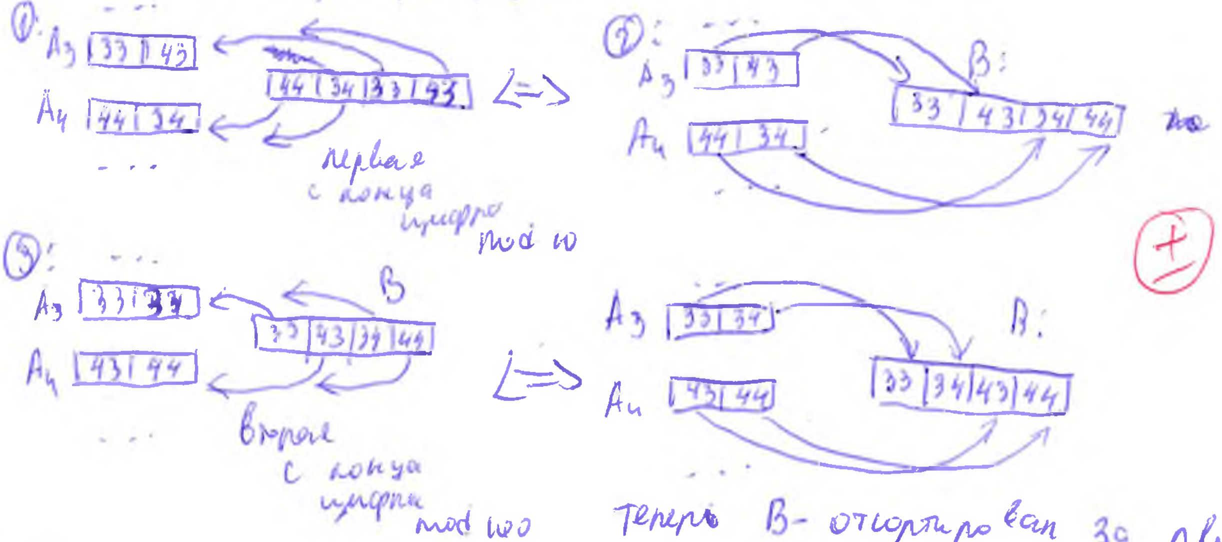
1. Проверяем, что элементы массива – цифры.
2. Сортируем таким эффект. образом 1-ую стопку, потом 2-ую, но записываем в обратном порядке, т.е. от max к min. И третью от min к max. Отлично, теперь все стопки отсортированы нужным нам образом.
3. Теперь сравниваем запомненные нами элементы, обозначенные 0, 1 и 2. Если новые значения в этих совпадают, то массив можно отсортировать. Если нет – нельзя и выводим сообщение (*).

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№3 (продолж.)

• Пример цифр. сортировки:

Пусть B = [44, 34, 33, 43]



теперь B - отсортирован за $n \log_{10} n$

±

Если нужно от большего к меньшему (т.е. $A_0 \ldots A_{10}$) (в задаче этот массив), то просто записать в обратном порядке.

№3

Так как a и b - веществ. числа и Серёжа очень много раз делил a:b:b:b:b ... b и получил ноль, то b>1.

Это произошло вследствии ограниченной памяти калькулятора. При получении ? числа, калькулятор имеет количество памяти для хранения числа, а веществ. числа (какие и получаются у Серёжи) округляются. Каждый раз при делении число получается всё меньше и меньше, а память - ограничена. Веществ. число всё больше приближается к нулю (0). В какой-то момент времени вещ. число ~~окр~~ будет настолько маленьким, что при округлении до какого-то порядка n в калькуляторе (чем больше памяти в калькуляторе, тем меньше порядок (т.е. до сотых до тысячн...) округл. => тем точнее знач. окр. чисел.) получается ноль.

Компьютер - ЭВМ, в котором памяти для хранения вещ. чисел больше, чем в калькуляторе. Значит, порядок округления m меньше (к примеру n - сотые; m - десятитысячные). Отсюда, округлённое число на компьютере точнее, меньше округляется => больше операций деления потребуется для того чтобы памяти для хранения точного рез-тата не стало хватать и округлённое число будет равно 0.

+

N 4

Рассмотрим матрицу N*N.

N - нечётное

	1	2	3	4	5
1	·	·	·	·	X
2	·	·	·	X	X
3	·	·	X	X	X
4	·	·	·	X	X
5	·	·	·	·	X

N - чётное.

Аналогично, только p_1 при N-чётном будет $\frac{N}{2}$ (вкл.) и p_2 - $\frac{N}{2}$ (невключая)

модель просмотра матрицы.

p_1 - параметр 1.

строки: i от 1 до $\frac{N+1}{2}$ включ;

столбцы: J от 1 до N-i вкл:

операции

k=1

p_2 - параметр 2

```
строки: i от N до $\frac{N+1}{2}$ невкл. в обр. порядке (шаг = -1):
    от J: 1 до N-k вкл.
        операции
    K = K+1
```

Для суммы - просто сложение.

Max элем - сравнение каждого max.

! i - строки; J - столбцы. A[i][J] - знач. матрицы в i, J.
pr - max знач элемента.

Алгоритм:

```
1. Ввод матрицы NxN (A - двумерный массив)
2. Sum = 0; pr = A[1][1]
3. Если N - нечётное, то:
    от i=1 до $\frac{N+1}{2}$ (включая):
        от J=1 до N-i (включая):
            если A[i][J] - не число: вывод „ошибка" и прекращ. работу программы
            Sum = Sum + A[i][J]
            ~~[illegible]~~
            Если A[i][J] > pr:
                pr = A[i][J]
    k = 1
    от i = N до $\frac{N+1}{2}$ (невкл, в обр. пор.):
        от J=1 до N-k (вкл.):
            если A[i][J] - не число - „ошибка".
            Sum = Sum + A[i][J]
            если A[i][J] > pr:
                pr = A[i][J]
        K = K+1
```

№4 (продол.)

```
4. Если N - чётное, то:
      от i=1 до $\frac{N}{2}$ (включая):
          от J=1 до N-i (включая):
              если A[i][J] - не число - „ошибка"
              Sum = sum + A[i][J]
              если A[i][J] > pr
                  pr = A[i][J]
      K=1
      от i = N до $\frac{N}{2}$ (не вкл., в обр. порядке):
          от J=1 до N-K (вкл.):
              если A[i][J] - не число - „ошибка".
              Sum = Sum + A[i][J]
              если A[i][J] > pr
                  pr = A[i][J]
          K = K+1
5. Вывести: print(sum,'- сумма');
6. Вывести: print(pr,'- max элемент').
```

±

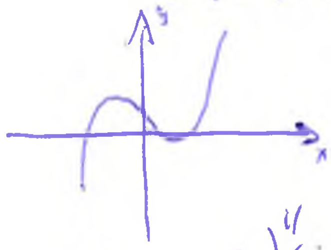
№5

$F(x)= 6(x^3+7x^2+3x+4) \bmod 10.$

∓

Пусть число $\overline{y_1y_2\dots y_n}$ преобразуется в $\overline{z_1z_2\dots z_n}$, тогда, чтобы $\overline{z_1z_2\dots z_n}$ преобразовывалось обратно в $\overline{y_1y_2\dots y_n}$, должен быть один вариант перехода, т.е. каждому элементу y_1, y_2, y_n соответств единств значение z_1, z_2, z_3 и наоборот.

То есть график, обратный F(x) тоже должен быть функцией.



не подходит

подходит

$\Downarrow$

~~функция монотонна! и непрерывна~~

$\Downarrow$

~~$F'(x) \geq 0$ и $F'(x) = 0$ при 1-ом значении x.~~

Вариант 73111

ШИФР НЕ ЗАПОЛНЯТЬ! ⇨ ДК 50-10.

~~$F_y'(x) = 3bx^2 + 7bx + 3b \geq 0.$~~

~~$3bx^2 + 7bx + 3b = 0$~~

~~$D = (7b)^2 - 4 \cdot 3b \cdot 3b \geq 0$~~

~~$\underline{(7b)^2 \geq 4 \cdot 3 \cdot 3 \cdot b^2 !}$~~

~~$\underline{49 \geq 36}$~~

~~Чтобы F(x) - монотонна, то $F_y'(x) \geq 0$~~

~~+ проверить x.~~

~~При этом - однозначное шифрование F(x).~~

~~[illegible] да~~

$F(x) = bx^3 \bmod 10 + 7bx^2 \bmod 10 + 3x \cdot b \bmod 10 + a b \bmod 10.$

$\Updownarrow$

F(x) должно быть различно при разных x.
(a, b - натур.)

```
1. a = 0
   b = 0
   kol = 0 kola = 0
2. Пока цикл не прервётся ~~[illegible]~~:            цикл A
       a = a + 1
       ~~от b = 1 до~~
       пока цикл не прервётся:                       цикл B
           b = b + 1
           l = [] - массив 0,
           ~~для~~ от x = ~~[illegible]~~ до x = ~~[illegible]~~ 9:
               k = ~~если~~ b·(x^3 + 7·x^2 + 3x + a) mod 10
               ~~[illegible]~~
               если k есть в l, то ~~выходим~~
                   kol = kol + 1
                   выходим из цикла в
               иначе: добавляем в массив
           если kol > 20:
               выходим из цикла.
       если kola > 20:
           выход из цикла.
3. Выводим a, и b.
```

Если при таких x не повторяется, то условие верно.

если есть b, который удовл. условие, то при kol выход >20 тогда в больше нет [illegible]

Олимпиада школьников «Надежда энергетики»

ИГЭУ

Место проведения

ИН70-62

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ ЕРШОВ

ИМЯ КИРИЛЛ

ОТЧЕСТВО ГЕННАДЬЕВИЧ

Дата рождения 24.04.2001 **Класс:** 11

Предмет ИНФОРМАТИКА **Этап:** ЗАКЛЮЧИТЕЛЬНЫЙ

Работа выполнена на 7 **листах** **Дата выполнения работы:** 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№3

Ноль получился потому что в памяти ~~компьютера~~ ~~[illegible]~~ ЭВМ числа хранятся в двоичном представлении, и на каждое число выделяется определенное количество бит. Из-за данных ограничений хранится только определенное число знаков после десятичной точки, и возникает погрешность. При делении число постоянно уменьшалось, пока не стало меньше погрешности. На компьютере используется более ёмкий тип данных, чем на калькуляторе, поэтому Сереже пришлось дольше уменьшать число, чтобы оно стало меньше погрешности (в более ёмком типе данных под число выделяется больше бит, что позволяет хранить больше знаков после точки и уменьшить погрешность.

№4

Таблицу на экране можно представить в виде двухмерного массива - matr[N, N]. Индексировать элементы будем с 1.

Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ СН-70-62

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
алг Задача4
цел N, i, j, K, sum, max, matr[N,N].
нач
  K=1
  sum=0
  ввод N, если N≤0, то [вывод: некорректные данные
                        [выход из программы.
  для i от 1 до N
  нц
    для j от 1 до N   {если в ячейке не записано число, то будем считать, что там −∞}
    нц
      ввод matr[i,j]
    кц
  кц
  max = matr[1,1]
  для i от 1 до N
  нц
  для j от 1 до N−K
   нц
    если matr[i,j]>max, то ?
        max = matr[i,j]
    всё если matr[i,j] ≠ −∞, то
    sum = sum + matr[i,j]
    кц всё
  если i < N div 2, то K = K+1 иначе
  если i = N div 2, то
       если N mod 2 ≠ 0, то K=K+1
  всё иначе K=K−1
  кц. если max = −∞, то вывод: нет чисел, иначе
  вывод: Сумма = sum, максимум = max
кон.
```

div – оператор целочисленного деления
mod – оператор взятия остатка от деления

±

Лист 02 из 04

№2.

Для расположения карточек будем использовать алгоритм быстрой сортировки. Ряды представим в виде массивов $P_1[N_1]$, $P_2[N_2]$, $P_3[N_3]$ целых чисел, проиндексированных с 1. N_1, N_2, N_3 – кол-во карточек в 1, 2, 3 рядах.

```
алг Сортировка (арг цел A[ ], l, r, c)
  нач цел q.
  если r > l, то
    q = разбиение (A, l, r, c)
    Сортировка (A, l, q, c)
    Сортировка (A, q+1, r, c)
  кон

алг цел разбиение (арг цел A[ ], l, r, c)
  ~~нач~~ цел i, j, m, k;
  нач
  i = l
  j = r
  m = (l + r) div 2
  пока i < j
    нц
  пока A[i]·c > A[m]·c нц i = i+1 кц
  пока A[j]·c < A[m]·c нц j = j-1 кц
  если i ≥ j, то выход из цикла кц.
  иначе
```

c = 1 – сорт. по возр.

c = -1 – сорт по убыв.

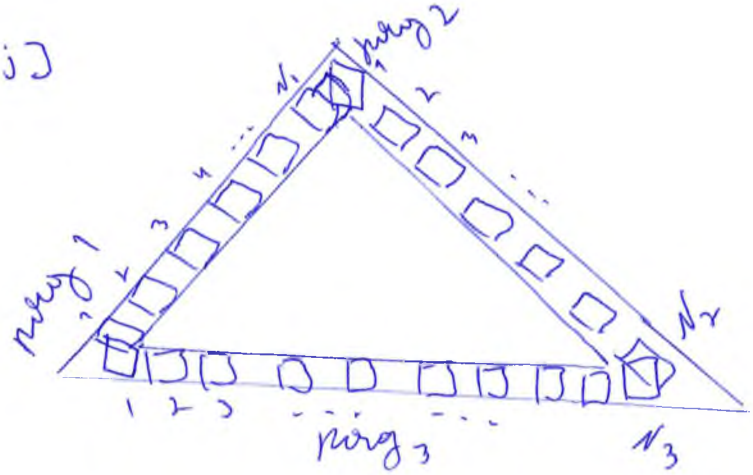
Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ 4Н70-62

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
K = A[i]
A[i] = A[j]
A[j] = K
i = i+1
j = j-1
всё
кц
вернуть j
кон.
```



```
алг Задача 2
  цел $N_1$, $N_2$, $N_3$, i, $P_1[N_1]$, $P_2[N_2]$, $P_3[N_3]$
нач
ввод $N_1$, $N_2$, $N_3$
если $N_1 \leq 0$ или $N_2 \leq 0$ или $N_3 \leq 0$, то
вывод("некорректные входные данные
всё
иначе
  для i от 1 до $N_1$ нц ввод $P_1[i]$ кц
  для i от 1 до $N_2$ нц ввод $P_2[i]$ кц
  для i от 1 до $N_3$ нц ввод $P_3[i]$ кц.
Сортировка($P_1$, 1, $N_1$, 1)
Сортировка($P_2$, 1, $N_2$, -1)
Сортировка($P_3$, 1, $N_3$, -1)
~~Следующий алгоритм используется для определения~~
```

Лист 04 из 07

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
для i от 2 до N_3 -1
  нц
  если P_3[i] = P_3[1] ~~или P_3[i] = P_3[N_3]~~, то
Вывод: на стыке не ~~[illegible]~~ минимальное значение
  всё
если P_3[i] = P_3[N_3], то
  Вывод: на стыке не максимальное значение
  всё
кц
всё
КОН.
```

№1

Воспользуемся тем, что (a·b)(mod K) = = a(mod K) · b(mod K) и будем при возведении в степень брать число по модулю 10. (Это последняя цифра).

```
Алг Задача 1
   нач цел fact, i, count
  i = 2
 fact = 1·2·3·4·5·6·7·8·9·10
  count = 1
пока count ≤ fact
 нц
 если степень(i, i) = 1, то
  вывод count
  выход из ~~цикла~~ ~~цкл~~ программы
  всё
 count = count + 1
   i = i + 1
```

```
  кц
вывод: период превысил 10!
  кон.
```

Опишем алгоритм бинарного возведения в степень по модулю 10.

```
алг цел степень (~~цел~~ арг цел n, k)
  нач цел res, mn
  res = 1
  mn = n
  пока k > 0
  нц
  если k mod 2 = 0, то
  mn = (mn · mn) mod 10
    res = (res · mn) mod 10
   k = k div 2
   в-сё
  иначе
  res = (res · mn) mod 10
   k = k − 1
  все
   кц.
   вернуть res
   кон
```

4

№5

~~Преобразование допускает однозначное расшифрование тогда и только тогда, когда F(x) ∈ [0; 9]. Это значит, что a принимает значения от~~

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ 15470-62

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Преобразование допускает однозначное расшифрование, когда для всех цифр X ~~F(X)~~ $F(X) \in [n; n+9]$, где n - некоторое число кратное 10.

±

Пусть ~~[illegible]~~ N - ~~[illegible]~~ кол-во цифр в числе.

~~Представила цифры в виде~~

Прочитаем последовательность цифр и запомним уменьшенный массив R.

```
для i от 1 до N нц ввод X, R[i] = $x^3+7x^2+3\cdot x$ кц
для b от 1 до 10
нц
для a от 1 до $9^3+7\cdot 9+3\cdot 9$
 нц ~~[illegible]~~ yes = ~~false~~ true
 для i от 2 до N    min = (R[1]+a)·b
  нц.
   если (R[i]+a)·b < min, то min = (R[i]+a)·b
  кц z = (min div10)·10
 для i от 1 до N нц
   если (R[i]+a)·b > ~~min~~ z+9 ~~или (R[i]+a)·b < z~~
   то yes = false
      выход из цикла
        всё
 кц
 если yes = true, то вывод: a, b
        выход из программы
        всё
 кц
 кц
 кон
```

Олимпиада школьников «Надежда энергетики»

МБОУ „СОШ №2" г.Гусь-Хрустальный

Место проведения

UN 48-63

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Козлов

ИМЯ Виктор

ОТЧЕСТВО Николаевич

Дата рождения 23.10.2000

Класс: 11

Предмет Информатика

Этап: Заключительный

Работа выполнена на 8 **листах**

Дата выполнения работы: 17.02.19
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

№4

~~1) Создаём матрицу Nx~~

1) Для удобства я буду обозначать элементы, находящиеся в матрице с 0.

2) Т.к только в некоторых клетках таблицы записаны целые число, то вводить я буду число и его координаты:

ввод: N, n i j; где n - целое число, N - размер таблицы, i - номер строки, j - номер столбца

3) Возьмём целочисленные переменные sum=0, для подсчёта суммы элементов, и maxn=-∞ (очень маленькое число), так как нам неизвестно, есть ли числа в запрещенной части таблицы.

4) во время ввода элементов мы будем отсеивать те, которые неподходят под условие (находятся в запрещенной части), следующим алгоритмом (в котором я буду находить и сумму элементов и maxn):

```
если i< (N+1)div2 то:
|    если j<N-i-1 то:
|    |
```

±

```
            sum=sum+n
            если maxn<n то:
                maxn=n
иначе:
    если j<i то:
        sum=sum+n
        если maxn<n то:
            maxn=n
```

5) После алгоритма мы должны вывести значения sum и maxn следующим алгоритмом:

```
если maxn==-∞ то:
    вывод: „Чисел в запрещенной области нет"
иначе:
    вывод: sum
    вывод: maxn
конец.
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

б) Отдельно в начале программы следует рассмотреть случай при N=1, если ввод чисел был, то нужно два раза вывести число n, то есть:

вывод: n

вывод: n

иначе вывести, что чисел в закрашенной области нет.

№1

1) т.к мы ищем %10 (mod 10), то нам не нужны сами числа, а только их окончания (иначе говоря, наши числа могут получиться при умножении числа само на себя несколько раз) Составим таблицу

число / степень	0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8	9
2	0	1	4	9	6	5	6	9	6	1
3	0	1	8	7	4	5	6	3	8	9
4	0	1	6	1	6	5	6	1	6	1
5	0	1	2	3	4	5	6	7	8	9

Видно, что при n^n, где n mod10 = 0, 1, 5, 6 всегда получается на конце n mod10

Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ VN 4Б-63

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Можно составить такую последовательность:

$1\,6^{8^{4^{2}}}\,1^{7^{9^{3}}}\,6^{4}\,5\,6\,1^{3^{9^{7}}}\,6^{8}\,1^{9}\,0$

④

Очевидно, что числа, у которых на конце стоят числа 4 и 8 - чётные => степень чётная => можно заменить на последовательность: $1\,6^{8^{4^{2}}}\,1^{7^{9^{3}}}\,4\,5\,6\,1^{3^{9^{7}}}\,8\,1\,0$ (а 9 - нечётные, поэтому степень нечётна)

Т.к 2 - чётное число => степень чётна, а 3 и 7 - не чётные => степень нечётна, то можно заменить на последовательность:

$1\,6^{4}\,7^{3}\,4\,5\,6\,1^{9}\,8\,1\,0$ => нам нужно смотреть на числа: 2; 3 7 → 12; 13 17 → $2^{2}\,2^{3}\,2^{7}$ → $3^{2}; 3^{3}; 3^{7}$ и т.д,

но, видно что 2 mod 4 = 22 mod 4

12 mod 4 = 32 mod 4

13 mod 4 = 33 mod 4

17 mod 4 = 37 mod 4

3 mod 4 = 23 mod 4

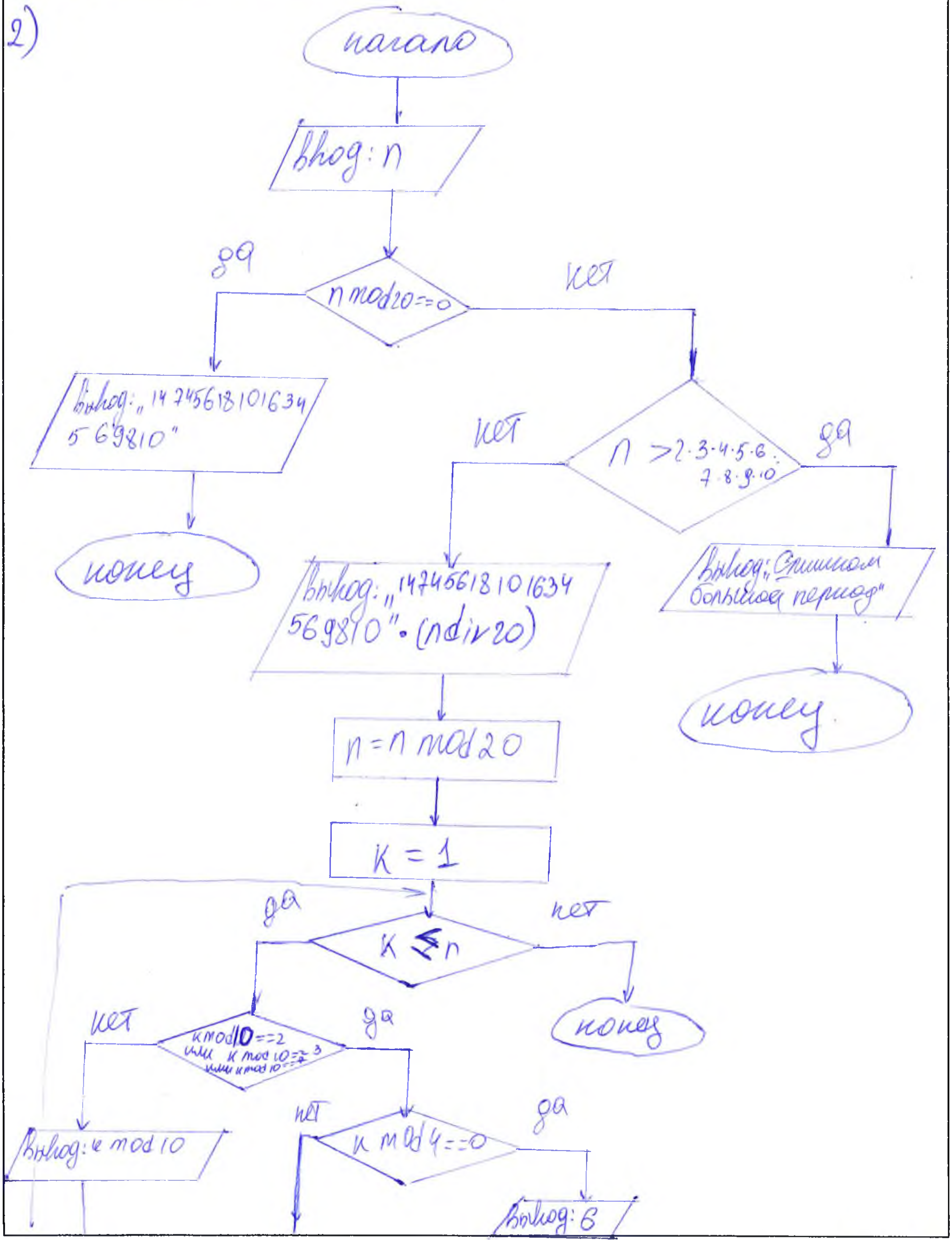
7 mod 4 = 27 mod 4

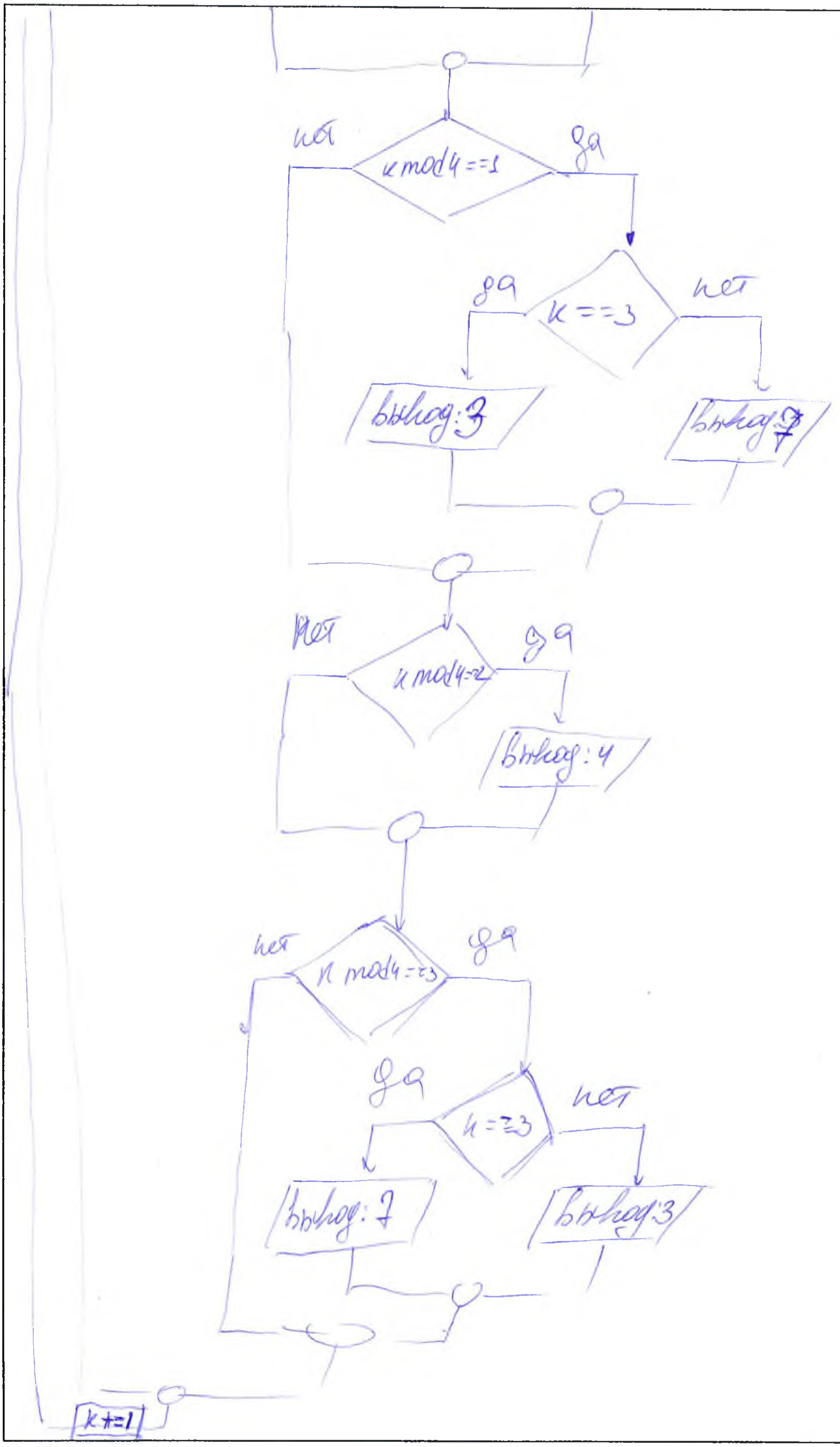
~~можно сделать вывод, что наименьший период при n≥20 будет ………,~~

Лист 4 из 8

можно сделать вывод, что последовательность 14745618101634569810 - будет повторяться

2)




нет
k mod4 == 1
да
да
k == 3
нет
вывод: 3
вывод: 7
нет
k mod4 == 2
да
вывод: 4
нет
k mod4 == 3
да
да
k == 3
нет
вывод: 7
вывод: 3
k += 1

№3

Так как числа вещественные, то ПК представляет их как 1,245 .... и.т.д, но до определённого момента. (Он берёт сколько-то чисел после запятой для конкретного типа данных), очевидно что при делении на число $b>1$ число a будет уменьшаться. В конце концов, т.к мы делим на число $b>1$, число a достигнет своего минимума в виде 0,0000.....n, где n какое-то число. При очередном делении кол-во нулей прибавиться, а памяти для числа k (число которое получилось) уже не будет => компьютер возьмёт часть с нулями, а k отбросит

$$\underline{0,0000\ldots a}k$$
⇓
взятое число. Поэтому число $a=0$.

Т.к. в калькуляторе меньше памяти на запоминание, то кол-ь получиться быстрее, т.к быстрее кол-во нулей перевалит за максимум.

(+)

(±)

№5

$b\cdot(x^3+7x^2+3x+a)$ 0/010 для каждого из 10-ти различных x, т.к цифр всего 10, нужно получить 10 цифр, чтобы допускалось одно значное декодирование

Очевидно, что b не может быть чётным, т.к при умножении будут получаться только чётные числа. А весь спектр ~~чисел~~ получается только при b = 9 или цифр

b = 1 или b = 7 или b = 3

Для этих чисел мы будем перебирать значения a, такие что: при подстановке x от 0 до 9 вычитая значение $(b\cdot(x^3+7x^2+3x+a))\bmod 10$ из Sum = 45 (сумма всех цифр) у нас получится ноль и никакая цифра не встречалась дважды. Для этого заведём ~~т~~ переменную массив на 10 элементов, который нумеруется с 0. Будем ~~вы~~ заменять (−1) значения для $i = (b\cdot(x^3+7x^2+3x+a))\bmod 10$. Если на каком-то из этих элементов не стоит (−1), то присваиваем значения массива их номера и меняем b, а если b = 9, то меняем a = a+1, а b = 1 и так далее, пока не подберём нужный элемент

№2

Создать массив, и сортировать элементы от 0 до k, от k+1 до n, от n до m, (где k – длина 1 ряда, (n − k + 1) – длина второго ряда, и (m − n + 1) – длина третьего ряда) разными способами и проверять стыки

∓

Олимпиада школьников «Надежда энергетики»

МЭИ, Москва

Место проведения

Ал 32-25

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73101

ФАМИЛИЯ ЛАПТЕВ

ИМЯ АНАТОЛИЙ

ОТЧЕСТВО АЛЕКСАНДРОВИЧ

Дата рождения 27.06.2002

Класс: 10

Предмет ИНФОРМАТИКА

Этап: ЗАКЛЮЧИТЕЛЬНЫЙ

Работа выполнена на 5 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Задание №1

1) Создадим пустой список "W" (для целых чисел)

2) пройдемся по значениям для x от 10 до 20, для y от 10 до 20, для z от 10 до 20 (влож. циклы) и для каждой комбинации x, y и z будем смотреть на значение выражения 3*(x*x)−y*y−7*z, если это значение равно 99, то заносим кортеж (x,y,z) в список W. +

После просмотра всех влож. циклов мы получим список W, который сод. кортежи (x,y,z) подход. комбинаций

Задание №3

После того как Серёжа n раз поделил на b, он получил значение $\frac{a}{b^n}$ на калькуляторе, и при ~~больш~~ $b^n >> a$ это число настолько мало, что становится меньше погрешности округления калькулятора, грубо говоря в десятичной записи этого числа после запятой нулей больше, чем входит на дисплей и калькулятор округляет это число до 0, т.е. $\boxed{0,0000} \Rightarrow \boxed{0}$ +

т.е. $\frac{a}{b^n}$ при больш. $n \to 0$

Задание №4

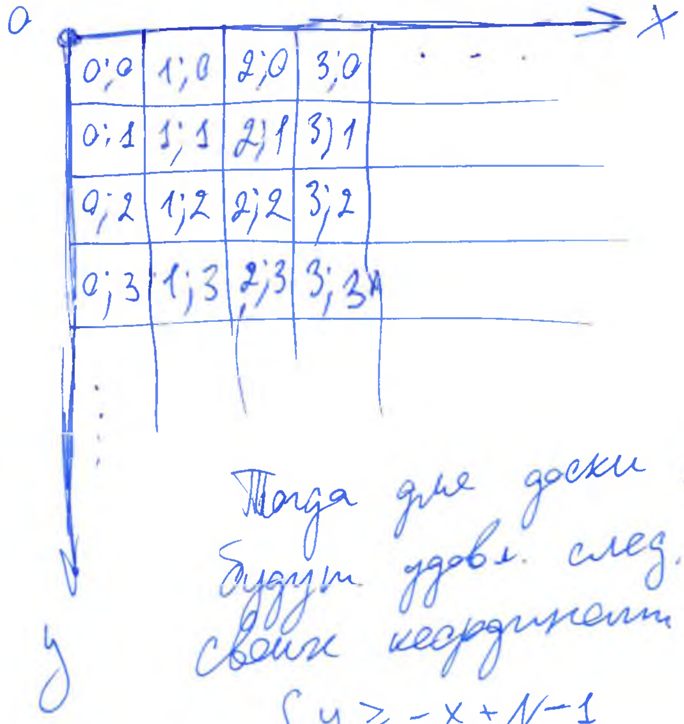
В этой задаче нам нужно научиться определять закрашена клетка или нет зная N и (x;y) (коорд. клетки) Пусть коорд. клетки отмечаются в след. системе отсчёта:

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ АИ 32-35

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



Тогда для доски N×N не закр. клетки будут удовл. след. системе уравн. для своих координат:

$$\begin{cases} y \geq -x+N-1 \\ y \leq x \end{cases}$$

```
Алгоритм:
  создаём пустой ~~стек~~ массив w (одном., целочисл.)
  считываем N
  считываем двум. массив N×N
  проб. по знач. x от 0 до N-1
        по знач y от 0 до N-1
            если не(($y \geq -x+N-1$) и ($y \leq x$)), то
                добавл. ~~(x,y)~~ значение этой клетки
                в ~~конец~~ массив w
  созд. цел. числ. перем. a = ~~-999999~~ w[0]
  ~~пробегаемся~~
  созд. цел. числ. перем. sum = 0
  пробегаемся по w (для i от 0 до (len(w)-1))
                                  длина w - 1
      sum += w[i]
      если w[i] > a то a = w[i]
```

±

мы получим перем. sum, значение которой - сумма чисел в закраш. клетках и перем. a, знач. которой макс. зн. из тех что были в таблице.

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ АИ 32-35

Задание №5

Для начала заметим, что если при a и в, ост. от деления на 10 которых m и n соотв.-но шифрование однозначно, то однозн. шифров. будет для всех a и в, у котор. остатки от дел. на 10 m и n, т.е. вместо возм. a и в мы можем искать m и n.

7

w - список (пустой)

проб. по значениям ~~а~~ m от 0 до 9, по знач. n от 0 до 9, ~~и значен~~ и создаём множество (пустое) k. Для всех a и в пробег. по x от 0 до 9, и для кажд. x добавл. в k значение ~~вы~~ `(n·(x*x*x+ 7*x*x+ 3*x +m)) % 10` если ~~то~~ после просмотра всех x для данных a и в k = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, т.е. ни одно значение не повторилось, то добавл. ~~(a, в)~~ (m, n) кортеж в список и мы получ. список кортежей (m, n), остатков от дел. a и в на 10, для котор. усл. выполн. И теперь мы знаем, что усл. будет выполн. для a и в в том случае, когда их остатки от дел. на 19 m_1 и n_1 будут в этом списке (списке подход. остатков)

Задание №2

Для выполн. пост. задачи нам потреб. ф-ия сортировки, давайте напишем её.

1. принимаем одном. массив и цел. число a, кот. будет обозначать в каком порядке (по возр. или по убыв.) нужно сортировать, пусть если a == 1, то по возр,
если a == -1, то по убыв.

```
если a == 1 :
    N = длина массива w (w - перед. массив)
    для i от 0 до (N-2)
целоч. перем. min = w[i]
        → d = i
        для j от (i+1) до (~~длина~~ N - 1)
            если If w[j] < min то (d = j; min = w[j])
        {меняем местами значения w[i] и w[d]}
        (мы пост. минимал. число из списка правее на i место)
так мы ~~[illegible]~~ заполн. всех. числа списка слева направо, замен. минимал. из оставш.)
    получаем отсортированно возр. список w

если a == -1 :
    N - длина массива w (w - перед. массив)
    для i от 0 до (N-2)
        max = w[i]
        d = i
        для j от (i+1) до (N-1)
            если w[j] ≥ max то (d = j; max = w[j])
        меняем местами знач. w[i] и w[d]
        (ставим max число из списка правее i на i место)
    получ. отсортиров. по ~~возр~~ убыв. список w
```

Метод называется „у картинки в музее", т.к. перед картиной стоят люди разного роста и мы ставим ~~вперед~~ на 1 самого низкого из тех кто ~~стоит~~ стоит с 1 по N место, на 2 самого низк. - из тех кто со 2 по N место. Это сорт. по возраст. не убыв. иначе, не выбирается самый высокий.

Итак, перейдём к задаче.

Теперь задача тривиальна.

сортируем первый ряд по возрастанию, второй по убыванию, смотрим на первое знач. в 3 ряду и последнее во втором (они же 1 и посл. в 3 ряду)

по уже изв. схеме наход. min и max 3 строки ~~если если min меньше~~ не совпад. с 1 и посл. числом или с посл. и перв. числом пишем (min и max не совпали с первон. значениями) иначе проводим сортировку этого ряда по возр. если 1 число - min, и по убыв. если 1 число - max

±

Олимпиада школьников «Надежда энергетики»

Москва

Место проведения

ЯК50-32

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Лукашов

ИМЯ Сергей

ОТЧЕСТВО Александрович

Дата рождения 18.01.2002 **Класс:** 11

Предмет Информатика **Этап:** Заключительный

Работа выполнена на 4 **листах** **Дата выполнения работы:** 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

№1

Заметим, что вычисление последней цифры числа n^ можно описать так:

$$((n\%10)(n\%10)\%10)(n\%10)\%10\ldots, \text{т.к. } (n\%10)\cdot(n\%10)\%10=(n\cdot n)\%10,$$

где a%b - остаток от деления a на b.

Значит за O(n) мы можем найти последнюю цифру числа n^n. не беспокоясь о переполнении ~~типа данн~~ переменной.

Алгоритм: (++x - увеличение x на 1) (a.append(t) - добавить t в конец массива a) break - выход из цикла;

```
r(x) {
    ans=1;
    for(i=1; i≤x; ++x){
      ans=(ans·(x%10))%10;
    }
    return ans;
}

main(){
  a=[], found=false, max_size=10!, i=1;
  while (a.size() < max_size·2){
     a.append(r(i));
     if (i%2 ==0){
        if(первая половина a равна второй половине a){
           found=true;
           break;
        }
     }
     ++i;
  }
  if (found){ print (a.size()/2);}
  else{ print ('Период не найден или он больше 10!');
}
```

Пр-ма!

±

Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ДК50-32

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№2

Назовём первый ряд A и в нем a карточек, второй - B с b карточками, третий C с c карточками. Введём нумерацию карточек от ~~1~~ где с 1 до числа карточек в ряду. Тогда

$A[1]=C[c]$, $A[a]=B[1]$, $B[b]=C[1]$. Поскольку карточка не может выйти из своего ряда, то сначала расставим карточки в рядах A и B, а потом постараемся расставить их в ряду C. Поставим на место $A[a]$ максимальную карточку из рядов A и B. Теперь мы можем отсортировать карточки в ряду A по возрастанию, а в ряду B по убыванию так, что это удовлетворяет поставленным условиям расположения карточек при обходе (A->B->C). Теперь отсортируем ряд C со 2-ого по c-1 элемент включительно по ~~[illegible]~~. возрастанию, чтобы карточки шли от минимального к максимальному. Теперь, если $C[c]<C[1]$ или $C[1]>C[2]$, то выводим сообщение о том, что значения на стыках не удовлетворяют условию. Иначе мы расставим карточки так как нужно. +

№3.

Если Серёжа получил 0 после нескольких операций, ~~то~~ ~~[illegible]~~ значит мы имеем дело с вещественными числами. Вещественные числа в памяти калькулятора хранятся в двоичном виде, но с ограниченной точностью. Поэтому, при ~~[illegible]~~ делении числа a на b, а потом умножении числа b на результат, Серёжа не получил числа a. Значит при попытке представления вещественного числа калькулятор округлил число до той точности, которую он может хранить. Поэтому после нескольких циклов „округления" числа, Серёжа получил 0. ~~[illegible]~~ На компьютере произошло то же самое. Только на компьютере вещественные числа представлены большим количеством бит, поэтому и их точность выше. но она всё равно не 100%, если ~~[illegible]~~ знаменатель дроби не является +

степенью двойки. Поэтому и на компьютере он получил 0, но сделал больше операций, чтобы преодолеть предоставленную компьютером точность.

№4

Введём нумерацию строк и столбцов с 0 до n-1 для таблицы n x n.

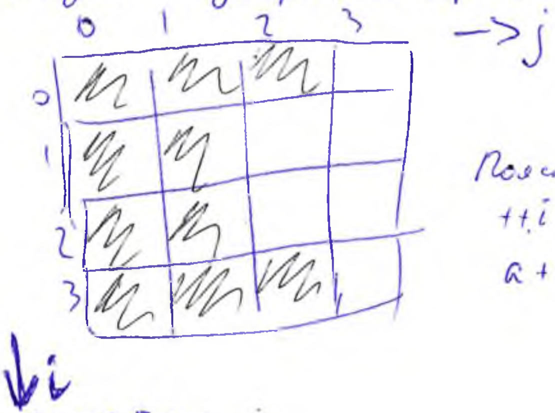


Таблица должна состоять минимум из 3-ёх строк/столбцов.

Пояснения к алгоритму:
++i - увеличение i на 1; --i - уменьшение i на 1;
a += b увеличение a на b.

алгоритм:

```
n, sum=0, max;
read(n);
if(n<3){ print('Неверное значение n')}
else{
  table[n][n];
  for(i=0;i<n;++i){
    for(j=0;j<n;++j){
      read(table[i][j]);
    }
  }
  max= table[0][0];
  for(i=0; i<n/2; ++i){
    for(j=n-i-2; j≥0; --j){
      sum += table[i][j];
      if(table[i][j]>max){ max= table [i][j]; }
    }
  }
  for(i=n-1; i≥n/2; --i){
    for(j = i-1; j≥0; --j){
      sum += table[i][j];
      if(table[i][j]>max){max=table[i][j];}
    }
  }
  print('Сумма элементов в закрашенной области: ', sum, ' Максимальный элемент: ', max);
}
```

±

№5

Предоставленное нам преобразование является полиномиальной хеш-функцией. Она не допускает расшифровку кроме полного перебора. Единственное, что мы можем проверить - это отсутствие коллизий на определенном наборе значений. Заметим, что значение заменяется на значение функции, взятое по модулю 10. Это значит, что у нас есть всего 10 уникальных значений. Значит однозначное сопоставление элементов на наборах, в которых количество уникальных значений больше 10 просто невозможно. Опишем алгоритм поиска подходящих a и b.

Так же заметим, что и a и b будут взяты по модулю 10, поэтому случаи, когда $a_1\%10=a_2\%10$, но $a_1 \neq a_2$ рассматривать бессмысленно. Значит a лежит в промежутке [0;9], и b [0;9] (только при b = 0 у нас все значения станут 0, но для одного элемента мы однозначно определим)

алгоритм: (++a - увеличение a на 1, append(t) добавление t в конец массива a)

```
F(x, a, b){
    return b·(x·x·x + 7·x·x + 3·x + a);
}

main(){
    x=[]; q=0;
    read(x);
    for(a=0; a≤9; ++a){
        for(b=0; b≤9; ++b){
            ciphered=[]
            for(t in x){ ciphered.append(F(t,a,b)%10); }
            if(для всех пар i,j   x[i]=x[j]⇔ciphered[i]=ciphered[j] и x[i]≠x[j]⇔
              ⇔ciphered[i]≠ciphered[j]){
                print('Возможно однозначное сопоставление при a= ',a,' b= ',b);
                print(конец строки);
                ++q;
            }
    } }
    if(q==0){ print('однозначное сопоставление невозможно'); }
}
```

±

Олимпиада школьников «Надежда энергетики»

СФМЭИ

Место проведения

GЯ 60-86

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73101

ФАМИЛИЯ Мащенко

ИМЯ Андрей

ОТЧЕСТВО Дмитриевич

Дата рождения 21.03.2003

Класс: 10

Предмет Информатика

Этап: Заключительный

Работа выполнена на 07 **листах**

Дата выполнения работы: 17 февраля 2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ G1 6086

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

N1. Для решения данной задачи достаточно перебрать только x и y в диапазоне от 10 до 20, а z вычислить по формуле

$$z = \frac{(3\cdot x\cdot x - y\cdot y) - 99}{7}.$$

⊕

```
начало{
  ~~переберем~~ перебираем x (от 10 до 20 включительно)
   начало
     перебираем y (от 10 до 20 включительно)
     начало
        int z = $\frac{(3\cdot x\cdot x - y\cdot y) - 99}{7}$; вычисляем z.
        if если (((3·x·x - y y )-99)%7==0) то начало
(обязательно проверяем, будет ли z целым, ~~ибо~~ только
//в случае, что z - целый, ~~мы и~~ имеет смысл рассматривать
// z как возможный ответ)
           ~~начало~~ (проверяем, удовлетворяет ли z условию)
           если ( z ≥ 10 и z ≤ 20)
            то начало
              вывод (x, '␣', y, '␣', z, '␣');
              вывод (перенос на новую строку)
            всё
   всё
     всё
   всё
конец.
```

2) Начало:

вводим массивы чисел, которые являются сторонами треугольника, и ~~[illegible]~~ их размеры.

```
~~ввод (~~ int razmerA=0;
    ввод( razmer A), int a[razmer A];
    ~~ввод (~~
    перебираем итератор (i) от 1 до razmer A.
        ввод (a[i]);
  int razmer B =0;
ввод (vazmer B); int b [vazmer B].
  Перебираем итератор (i) от ~~1~~ 1 до razmer B
    ввод (b[i]);
int vazmer C=0
ввод (razmer C); int c[razmer C];
  Перебираем итератор (i) от 1 до razmer C
    ввод (c[i]).
если (razmer A равен 0 или vazmer B равен 0 или vazmer C равен 0) то
    начало
      вывод (a, "Некорректный ввод");
    конец.
    все
```

```
Сортируем по возрастанию массив a:
  перебираем i от 1 до razmer A - 1 ~~n-1~~ включительно        ?
    перебираем j от 1 до razmer A-i [illegible] включительно
      начало:
        если (a[j]>a[j+1]) то начало
          int x = a[j+1]; a[j+1]=a[j]; a[j]=x;
        все
      все
```

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ СЯ 60-86

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
сортируем по убыванию массив b;
  перебираем i от 1 до razmerB-1 включительно
    перебираем j от 1 до razmer B-i включительно
    начало
        если (b[j]<b[j+1]) то начало
          int x= b[j+1], b[j+1]=b[j]; b[j]=x;
          всё
    всё;
```

присваеваем последнему элементу массива а первый элемент массива b, так как мы двигаем ~~[illegible]~~ углы треугольника по мере его прохождения,

a[X]=b[1],

```
сортируем по убыванию массив c:
  перебираем i от 1 до razmer C-1 включительно
    перебираем j от 1 до razmer C-i включительно
      начало
        если (c[i]< c[i+1])то начало;
          int x= c[i+1], c[i+1]=c[i], c[i]=x;
          всё
        всё;
```

теперь найдём ~~минимальн~~ минимальный и максимальный элемент массива C. объявим их:

~~[illegible]~~ int max = -100000, int min= 100000,

~~[illegible]~~

```
перебираем ⓘ от 1 до razmer C включительно
начало
  если (c[i] > max) то max = c[i],
  если (c[i]< min) то min= c[i];
всё.
```

Теперь проверяем, совпадают ли максимальные элемент массива с с началом a и минимальный с конечным элементом массива b. Если нет, то ставим.

```
если (a[1] не равно max и b[razmerB] не
равно min) то
    начало
        вывод ("Несовпадение"),
            b [razmerB]=min, a[1]=max;
    все
конец.
```

±

Мы получили массивы a, b и c, соотв. сторонам треугольника, отсортированные по условию.

№3. Из-за многочисленного деления числа на одно и тоже, дробная часть ответа уменьшалась. Длины вещественного типа не хватило для хранения дробной части ответа и, когда в доступных ячейках остались нули, калькулятор ~~приравнял~~ приравнял ответ к нулю. ±

№4.

начало; ~~вв инт~~ объявление и ввод (n).

```
    int n; ввод (n);
  [если (n≤0) то начало вывод ("Некорректный ввод");
                          конец.
```

~~объявление~~

~~если (остаток от деления n на 2 равен 1) то~~
~~начало~~

объявление массива a (таблицы n×n) и временного массива, в котором будут храниться числа с закрашенной части таблицы, также заполним его ~~[illegible]~~ '-10000';

```
    int a [n][n];
    int zakr[n·n];
```

Олимпиада школьников «Надежда энергетики»

Вариант: 78101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ GЯ 60-68

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
Перебираем итератор i от 1 до n·n включительно
   zakr[i]=-10000;
int Zsize=0; (объявим ~~границу~~ правую границу массива zakr.
если n кратно 2, то таблица ~~получается~~ является симметричной, иначе – в середине будет асимметричная строка.
если (остаток от деления n на 2 равен 1) то
  начало
    int nedost=1; (недоступные справа в таблице ячейки)
    int it=1; (временная правая граница zakr)
    перебираем i (от 1 до (n-1)/2 включительно)
      начало
        перебираем j (от 1 до n-nedost включительно)
          начало
            zakr[it]= a[i][j]; ~~it++~~ it=it+1; (добавим в массив закрашенных чисел число из таблицы и сместим границу массива на 1 вправо).
            zakr[it]= a[n-i+1][j]; ~~it++~~ it=it+1; (то же самое делаем симметрично)
          все
  все  nedost=nedost+1; (увеличиваем недоступные ячейки справа в таблице на 1)
  Отдельно средняя линия в таблице.
   int i=(n+1)/2;
  Перебираем j от 1 до n-nedost включительно
     начало
        zakr[it] = a[i][j]; (добавим в массив закрашенных чисел ещё число из средней линии в трапеции)
        it=it+1; (смещаем границу вправо на 1)
   Zsize= it-1; (размер массива с числами (итоговый))
все.
```

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ GЯ 60-86

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
В случае, если n-четное, то таблица является симметричной.
если (остаток от деления n на 2 равен 0) то
  начало
    int nedost=1; (недоступные справа в таблице ячейки)
    int ct=1; (временная правая граница массива (закрашенными числами)
    перебираем i (от 1 до n/2 включительно)
      начало
        перебираем j (от 1 до n−nedost включительно)
        начало
          zakr[ct]=a[i][j]; ct=ct+1;
          zakr[ct]=a[n-i+1][j]; ct=ct+1;
        (добавляем в массив с числами закрашенные числа из таблицы)
        все
      nedost=nedost+1; (увеличиваем количество недоступных ячеек в таблице справа на 1.
    все
  zsize=ct-1; (итоговый размер массива с числами равен правой границе минус одну ~~[illegible]~~ лишнюю операцию ~~прибавления 1)~~ увеличения на 1)
Ищем максимальный, минимальный элемент и сумму.
  int mx=-10000; mn=10000; sum=0; (объявление)
  перебираем i от 1 до zsize включительно
    начало
      если (zakr[i]>mx) mx=zakr[i]; (максимум
      если (zakr[i]<mn) mn=zakr[i]; (минимум)
      sum=sum+zakr[i]; (сумма
  все
вывод ( ~~[illegible]~~ mx, '␣', mn, '␣', sum);
конец.
```

±

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ СЯ 60 - 86

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

±

```
№5. Начало:
    ~~ввод~~ int len=0; (объявление длины последовательности)
    ввод (len);
    Перебираем i от 1 до len включительно
      начало
        int x; (объявление x)
        ввод (x), если(x<0) то начало вывод('Некорректный ввод'),
                                 конец; всё;
        int t = x·x·x + 7·x·x + 3·x; (объявление t — значения выражения
                                                $x^3+7x^2+3x$).
        перебр a от 1 до ~~2000~~ 315 включительно
          перебр b от 1 до 315 включительно
    если(остаток от деления числа (b·(t+a)) на 10 равен x), то
          начало
            вывод ('Для данных a и b выполнено
    поставленное условие:', a, ' и ', b);
            конец переборы;
          всё
      всё
      ~~конец~~.
    Конец
```

Олимпиада школьников «Надежда энергетики»

СОШ №20

Место проведения

24 37-49

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73101

ФАМИЛИЯ Михайлова

ИМЯ Ольга

ОТЧЕСТВО Олеговна

Дата рождения 24.04.2002

Класс: 10

Предмет информатика

Этап: заключительный

Работа выполнена на 3 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ 2437-49

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№1.

Так как диапазон изменения переменных мал, и, следовательно, кол-во комбинаций для компьютера так же мало (11^3), то данную задачу можно решить перебором.

```
начало
x:=10
пока x≤20
начало
    y:=10
    пока y≤20
    начало
        z:=10
        пока z≤20
        начало
            если $3x^2-y^2-7z=99$ то выводим x';'y';'z ↲
            z:=z+1
        конец
        y:=y+1
    конец
    x:=x+1
конец
конец.
```

(+)

(∓)

№2.

В начале считываем значения карточек в каждом из рядов в отдельный массив. (1) После сортируем первый ряд по возрастанию и сохраняем значение наибольшего элемента в переменной max1. (2) Меняем значение первого элемента второго массива на max1 и сортируем второй ряд по убыванию, а затем сохраняем значение максимального элемента массива в переменной max2. В цикле повторяем шаги (1) и (2) пока max1 ≠ max2. После завершения цикла переменная min1 принимает значение первого элемента первого массива, а переменная min2 – значение последнего элемента второго массива. ? Значения первого и последнего элементов третьего массива заменяются на min2 и min1 соответственно. Затем, если min2 < min1, то сортируем третий ряд по возрастанию. Если первый элемент третьего массива равен min2, а последний – min1, то выводим все три массива как решение, иначе

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

выводим „Невозможно". Если же $mn2 \geq mn1$, то сортируем третий ряд по убыванию, и если первый элемент третьего массива равен mn2 и последний – mn1, то выводим все три массива, иначе выводим „Невозможно".

Считывание элементов массивов происходит так:

Сортировку можно проводим „методом пузырька":

```
начало
  i:=1
  пока i<n
  начало
     j:=n
     пока j>i
     начало
        если $a_j < a_{j-1}$ то   // - это сортировка по возрастанию;
        начало
           x:=a_i                 // по убыванию: $a_j > a_{j-1}$
           ~~$a_{i-1}$:=~~ a_i:=a_i-1

           a_i-1:=x
        конец
        j:=j-1
     конец
     i:=i+1
  конец
конец
```

№3.

Под каждое число в памяти калькулятора выделяется определенное кол-во памяти. Поэтому после каждого деления вещественных чисел a и b в памяти записывается не точное, а приближенное значения частного (для вещ. чисел - до n-го знака после запятой). Также на дисплее калькулятора ~~огр кол-во~~ ограниченное кол-во позиций для цифр, поэтому, когда частное от деления станет очень маленьким (для памяти калькулятора), то на дисплее выведется приближенное значение, равное нулю. +

№4.

Таблицу N×N можно представить как матрицу N×N. Каждый элемент матрицы – a_{ij}, где i – номер строки (начиная сверху), а j – номер столбца (начиная слева).

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

В начале с помощую вложенного цикла считаем часть элементов матрицы $a_{i,j}$, где i меняет значение от 1 до N div2+N mod 2, а j меняет значения от 1 до N−i (j меняет значения в цикле внутри цикла, в котором меняет значения i) Во время считывания увеличиваем переменную sum на $a_{i,j}$ (в начале программы sum:=0), а каждый элемент массива b_k присваиваем значение $a_{i,j}$ (в начале k:=1, после каждого присваивания k увеличивается на 1). После цикла повторяем вышенаписанное, только i меняет значения от ~~1 до N div 2~~ N div2 до 1, а j меняет значения от 1 до $\underset{i-1}{N-i}$, считывается же переменная $a_{N-i+1,j}$.

±

После сортируем массив b по убыванию, тогда b_1 равен максимальному элементу массива и закрашенной части таблицы. Выводим как ответ переменные sum и b_1.
(сортировка описана подробнее в №2)

№5.

В цикле находим значение многочлена x^3+7x^2+3x для целых x от 1 до 9 и записываем остатки от деления полученных значений на 10 в элементах массива d_{x+1}. После этого проверяем: если все элементы массива d_{x+1} различны, то a – любое натуральное число, а ~~b={1,3,7,9}~~ b может принимать значения ~~чисел~~ чисел, оканчивающихся на 1, 3, 7, 9; иначе невозможно подобрать такие a и b.

Различность элементов в массиве можно проверить так: отсортировать массив и проверить на различность каждую пару рядомстоящих чисел.

±

Олимпиада школьников «Надежда энергетики»

МБОУ «СОШ №4» г. Гусь-Хрустальный

Место проведения

UN 48-32

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Мишанин

ИМЯ Никита

ОТЧЕСТВО Алексеевич

Дата рождения 02.05.2001

Класс: 11

Предмет Информатика

Этап: Заключительный

Работа выполнена на 7 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Задача 1

Для начала опишем глобальную переменную строкового типа S, где s пустая строка

Напишем функцию которая будет записывать в строку наш возможный период. Назовем эту функцию f и она будет иметь строковый тип

f (число a, степень b) – передаем две числовые переменные где a = b

{ в самой функции

опишем переменные i, k, где k = a, и переменные z

запустим цикл

От (i = 2, пока i <= b, с шагом +1)

K = k * a

после цикла в k лежит наше число n^n,

берем от него остаток от деления на 10 и кладем в z, z = k % 10

теперь переведем z в символ и прибавим к строке S, для этого опишем переменную типа строк n, и переведем из числа в строку

n = строка от z, прибавим к S

S = S += n

~~вернем в основную программу~~.

нам не надо возвращать S из функции, т.к S – глобальное

функция закончилась

}

Основная программа

Возьмем переменные i, n! = 1·2·3·4·5·6·7·8·9·10, n = 0, где i = 1

Напишем цикл

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
пока (n <= n1)
    делаем следующие
    запускаем функцию f от (i, i)
    в n кладем m, где m - число взятое от строки (S) (перевод из строки в число)
    увеличиваем i на один
```

По окончанию цикла у нас есть строка S, где и лежит период. Идем по этой строке, Опишем две пустые строки S1 и S2

```
от (i = 0, пока i < длины строки S, с шагом +1)
    в S1 += элемент строки S (S[i])
    цикл
    от (j = i+1, пока j < длины строки S1, с шагом +1)
        в S2 += элемент строки S, но с индексом j (S[j])
Теперь проверим,
    если строки S1 и S2 равны, то S1 и есть минимальный период (проверку делаем в цикле)
```

Если по окончанию основного цикла мы не нашли период, заканчиваем программу

Задача 4

Пойдем от обратного, присвоим нашей незакрашенной части значение 0

Для этого пойдем двумя вложенными циклами

			n		
					0
				0	0
n			0	0	0
				0	0
					0

- как на примере

но сначала опишем переменные j, i, k, f, n

и наш массив a, размерностью [n, n]

Итерацию массива возьмем с 1 и до n включая

~~пока~~ от (j = n, пока j > 0; и у j вычитаем 1, то есть шаг (-1))

в этом цикле

от (i = k, пока i <= f, с шагом +1),

где k и f изначально 1 и n, k = 1, n = f

зануляем элемент

a[i][j] = 0

±

после этого цикла прибавляем k += 1,

вычитаем f -= 1

проверяем если (k > f)

выходим из главного цикла

Теперь нам осталось посчитать сумму и найти максимум

переменные

Sum = 0,

max = -2000000

большое отриц. число

от i = 1 i <= n, с шагом +1

от j = 1, j <= n, с шагом +1

Sum += a[i][j] - добавляем в сумму элемент массива a

если (a[i][j] > max)

max = a[i][j]

в итоге в переменной Sum - сумма элементов, в переменной max - максимум

Задача 2

У нас есть схема вида

ряды карточек и треугольником

Обозначим a, b, c - как ряды

есть несколько ситуаций обхода, когда

1) a - 1, b - 2, c - 3 (порядок обхода)

2) a - 3, b - 1, c - 2 3) a - 2, b - 3, c - 1

Для алгоритма рассмотрим первый вариант обхода

разобьем стопки на три динамических (вектор) массива a, b, c

где a – первая стопка (min → max)
b – вторая (max → min)
c – третья (min → max)

И так, у нас есть 3 массива, отсортируем их с помощью пузырьковой сортировки

```
от (i = 1, до i < n - 1, с шагом +1)
    от (j = n-1, j > i, с шагом -1)
        если (a[j] > a[j+1])
            меняем местами
```

Соответственно
Применив её мы получим три массива, два отсортированных по возрастанию, один по убыванию, для убывания мы после сортировки просто развернем массив, т.к будем использовать динамический его вид. посмотрим на их вид

a = [1] .... [n]
↑min ↑max

b = [1] ... [n]
↑max ↑min

c = [1] .... [n2]
↑min ↑max,

Чтобы проверить правильности осталось сверить

a[1] и c[1], так же a[n] и b[1] и b[n] и c[n].

если (a[1] = c[1], и a[n] = b[1] и b[n] = c[n])
    тогда все верно
иначе
    выводим сообщение об том, что на стыке не минимум и максимум.

±

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Задача № 5

∓

У нас есть функция вида $f(x) = b\cdot(x^3+7x^2+3x+a)$ есть цифры, которые мы кодируем

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Для каждой цифры найдем значение $f(x)$

$f(0) = ab$
$f(1) = b(11+a)$
$f(2) = b(42+a)$
$f(3) = b(99+a)$
$f(4) = b(188+a)$
. . .
и так до $f(9)$.

каждая цифра должна быть однозначно зашифрована а.т.к мы шифруем так, что берем от $f(x)$ остаток от деления на 10, то одинаковых остатков быть не должно само число не может кодироваться собой же, поэтому $a\cdot b$ не кратно 10

тем самым у нас есть 10 выражений, которые при делении на 10, должны дать разный остаток исходя из $a\cdot b \nmid 10$, мы знаем что ни a ни b не кратно 10, а цифру ноль нужно как то закодировать, значит какое то из выражений $(11+a)(42+a)$ ... и т.д кратно 10, тем самым мы можем ограничить наш алгоритм, для реализации я использую такую структуру как упорядоченное множество без повторяющихся элементов и назову его (a') в (a') мы будем складывать остатки, и если по итогу 10 выражений вида $f(x)$ ... мы получим размерность $a' = 10$, то это и будет ответ, выведем a и b. Реализация

Будем идти циклами и проверять значение выражений, нам достаточно делать это для a, так как мы будем проверять кратность для девяти функций, начиная с f(1)

от (a = 1 до a <= 100, с шагом +1)

Считаем 9 выражений вида $f(x^3+dx^2+3x+a)$

начиная с $x = 1$, например значения будут храни

в 9 переменных

$x, y, z, x', y', z', x'', y'', z''$,

теперь проверяем кратность каждой переменной на 1

если нашли первую попавшуюся, то вычисляем b и ищем (добавляем) остатки в множество

если $\underbrace{\text{множество}}_{\text{размер}}$ $a' = 10$, то вывод a'

a и b.

Иначе очищаем множество и делаем при большем a

Задача 3

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

выполнение операций с вещественными числами происходит по разному как на калькуляторе, так и на компьютере, связано это с тем, что компьютер задействует больше памяти, чем обычный калькулятор, а значит способен хранить большее количество знаков после плавающей точки, при делении калькулятор и компьютер отбрасывают незначительную часть, но у калькулятора эта часть больше и быстрее отбросы величя, тем самым быстрее доходя до нуля, поэтому нужно большее количество операций на компьютере, чем на калькуляторе, чтобы число вида

$c=a/6$; $c=c/6$, $c/6$ .... дошло до 0.

+

Олимпиада школьников «Надежда энергетики»

МБОУ «СОШ №2»
г. Гусь-Хрустальный

Место проведения

jD 51-67

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 23101

ФАМИЛИЯ МУХИН

ИМЯ ЕЛИСЕЙ

ОТЧЕСТВО АЛЕКСАНДРОВИЧ

Дата рождения 24.06.2002

Класс: 10

Предмет ИНФОРМАТИКА

Этап: Заключительный

Работа выполнена на ~~5~~ 6 **листах**

Дата выполнения работы: 17.07.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№1

```
начало алгоритма
 цел x, y, z, p
 нц по x от 10 до 20.
     нц по y от 10 до 20:
         p = 3x2-y2 -55
         z = $\underbrace{(3x^2-y^2-55)}_{p}$ div z
         если $\underbrace{(3x^2-\cancel{x}y^2-99)}_{p}$ mod z ==0 и p div z ≥10 и p div z ≤20, то
             x, y, z - искомая комбинация
     кц.
 кц
конец алгоритма
```

(+)

№3

Калькулятор не может хранить бесконечную десятичную дробь, поэтому он хранит определённое количество цифр после запятой, пусть это количество равняется k, тогда если в результате деления количество цифр после запятой больше k, то оставшиеся цифры отбрасываются и если результат деления имел вид $0,\underbrace{00..00}_{k}\underbrace{\dots}_{\text{какие-то цифры}}$, то после отбрасывания число станет равно $0,\underbrace{00..00}_{k}$ или просто 0.

±

№5

```
начало алгоритма
 цел x, t, a, b, i.
 лог flag
 массив целых чисел a на 10 элементов (массив нумеруется с нуля)
 нц по i от 0 до 9.
     a[i] =0
 кц
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Продолжение к номеру 5

±

```
ввод a, b
нц x от 0 до 9:
    t= b·(x^3+2x^2+3x+a) mod 10
    A[t] = A[t]+1
кц.
flag = True
нц i от 0 до 9:
    если A[i]>1, то
        flag = False
кц.
если flag == True, то
    узорное преобразование допускает однозначного расшифрования
иначе
    узорное преобразование не допускает однозначного расшифрования
конец алгоритма
```

Этот алгоритм служит для проверки констант a и b. После применения этого алгоритма для некоторых a и b можно понять закономерность и решить задачу.

N 4

```
начало алгоритма
цел t=1, sum=0, max, i, j, N, k
ввод N
двумерный массив целых чисел matrix на N×N элементов
нц. i от 1 до ~~N-1~~ N                  // нумерация в массиве matrix с единицы
    нц j от 1 до ~~N-1~~ N
        ввод matrix[i][j]
    кц
кц.
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Продолжение к №4

```
k = 1-N mod 2
max = a[1][1]
 нц i от 1 до N
    нц j от 1 до N-t ;
        Sum = sum + ~~[illegible]~~ matrix[i][j]
        если max < matrix[i][j]
            max = matrix[i][j]
    кц.
    если i < (N+1) div 2 , то
        t = t+1
    иначе
        если i > (N+1) div 2 + k , то
            t = t-1
 кц.
 если N = = 1, то
    Вывод „сумма и максимум не найден"
 иначе
    Вывод Sum, max
конец алгоритма.
```

±

№2

Алгоритм сортировки

```
функция qsort (массив цел чисел a , цел left , цел right){
    если left +1 < right , то
        l = left
        r = right
        x = (l+r)/2
     нц. пока r-l ≥ 1
            если a[l] > x и a[r] ≤ x , то
```

Продолжение к №2

```
        если a[l]>x и a[r]≤x, то
          tmp = a[l]
          a[l] = a[r]
          a[r] = tmp
          l = l+1
          r = r-1
        иначе
          если   a[l]>x
                r = r-1
          иначе
                если a[r]≤x
                   l = l+1
                иначе
                   r = r-1
                   l = l+1
        кц.

        qsort(a, left, r)
        qsort(a, l, right)

}
```

Отсортируем 1, 2, 3 ряд и проверим, что в ~~первом ряду~~ 1 и 3 ряду 1 элемент < 2 ого и последний элемент > предпоследнего, а во 2 ряду 1 элемент > 2 ого и последний элемент < предпоследнего и если условия не выполнились, то выведем сообщение об ошибке.

Т.к. 2 ряд нужно отсортировать по убыванию, то после qsort просто развернем его следующим алгоритмом

```
    нц. i от 1 до кол. элементов div 2
        ~~a[i]~~ tmp=a[i]
                 a[i] = a[кол элементов - i + 1]
                 a[кол. элементов - i + 1] = tmp
    кц.
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№2.

более медленный способ отсортировать массив.

```
цел i, j, tmp. , N - количество элементов в массиве a
 нц i от 1 до N:
     нц j от i+1 до N:
         если a[j] < a[i]:        // если в условии a[j] < a[i] поменять знак
             tmp = a[i]             < на >, то массив отсортируется по
             a[i] = a[j]              убыванию
             a[j] = tmp
     кц.
 кц.
конец алгоритма
```

±

–

№1

Альтернативный алгоритм

```
начало алгоритма
 цел x, y, z
 нц. x от 10 до 20.
     нц y от 10 до 20
         нц z от 10 до 20
             если 3x²-y²-7z ==59, то
                 x, y, z искомая комбинация
         кц
     кц.
 кц.
```

№2.

альтернативный алгоритм сортировки

```
 цел i, j, k, mn, tmp, N - кол элем. в массиве a
  нц i от 1 до N
     mn = a[i]
     k = i
     нц. j от i+1 до N
         если a[j] < mn
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
        если a[j] < mn, то
            mn=a[j]
            k=j
    кц.
    если k ≠ i, то
        tmp=a[i]
        a[i]=a[k]
        a[k]=tmp
кц.
конец алгоритма.
```

№3

Приведем пример: пусть дисплей отображает 5 цифр после запятой;

a=1 b=10

после 5 делений на b результат станет равен 0,00001, и когда Сергей попытается разделить на b ещё раз, то результат должен быть равен 0,000001, но т.к. дисплей не может отображать 6 цифр после запятой, то последняя 1 отбрасывается и результат оказывается равен 0,00000 или 0

№5

К продолжению № 5.

Чтобы расшифрование было однозначно число b должно оканчиваться на цифры {3 ; 7 ; 5}

Олимпиада школьников «Надежда энергетики»

СОШ №20

Место проведения

9Ч34-61

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Овчинников

ИМЯ Андрей

ОТЧЕСТВО Владимирович

Дата рождения 07.11.2000 **Класс:** 11

Предмет Информатика **Этап:** Заключительный

Работа выполнена на 4 **листах** **Дата выполнения работы:** 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Олимпиада школьников «Надежда энергетики»

Вариант: 73ш

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ЯУ 34-61

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

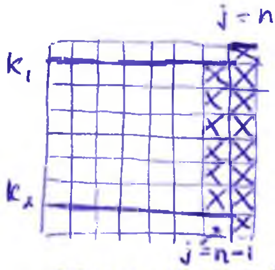
4. Сначала нужно заполнить таблицу :
(Элементы программы написаны на C++)

1)
```
for (i = 1; i <= n; i++)
for (j = 1; j <= n; j++)
cin >> a[i][j]
```

Заполняем двумерный массив различными числами
i – счётчик строк
j – счётчик столбиков

2) Затем позиции с числами в незакрашенных клеточках присвоим к нулю

```
j = n;
k1 = 1
k2 = n;
while (k2 ≥ k1)
{ for (i = k1; i <= k2; i++)
a[i][j] = 0;
k1 = k1 + 1;
k2 = k2 - 1;
j = j - 1; }
```



Сначала ~~так~~ занулим последний столбец, переместимся назад (j = j - 1) и сдвинем „границы зануления" k1 и k2, которые как бы стягиваются к центру таблицы с каждым новым шагом

Таким образом получим таблицу, в которой в незакрашенной части стоят нули, а значит можно просто сложить все элементы и мы найдём нужную сумму. Так как в задаче гарантируется что есть ненулевые целые числа, то „0" также не станет максимальным значением

```
for (i = 1; i <= n; i++)
for ( j = 1; j <= n; j++)
S = S + a[i][j]
```

Сложим все элементы массива → найдём сумму элементов в закрашенной части

Пр-ма! I

```
for (i = 1; i <= n; i++)
for ( j = 1; j <= n; j++)
if ( a[i][j] > max )
max = a[i][j]
```

Найдём максимальное значение в таблице

5. остаток деления числа на 10 – это цифра от 0 до 9 Однозначное расшифрование будет тогда когда каждой первоначальной цифре будет соответствовать одна единственная новая цифра

~~$F(0) = b \cdot (0^3 + 7 \cdot 0^2 + 7 \cdot 0 + a) = ba$~~
~~$b < 10$ так как при $b = 10$ остаток от~~

I

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ЯУ34-61

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
for (a=0; a<=n; a++)
for ( b=0; b<=n; b++)
{
for (i=0; i<=9; i++)
{x = b*(i*i*i+7*i*i+3*i+a);
s.erase(x, 1); }
if (s.lengh ==0)
{ break;
cout << a << " " << b;
}
} s = s0;
}
```

Перебираем значения для a и b в заданном диапазоне n (например до b = 1000 и т.д)

В теле цикла шифруем каждую из цифр в новую

Заранее создаем строку S_0 = "0123456789"

После шифрования вырезаем из строки S ($S = S_0$ в начале) полученный символ (номер символа в строке соответствует значению цифры)

Таким образом, если после шифрования всех цифр длина строки S будет равна 0, то это значит, что повторов при шифровании не было и мы нашли нужные нам a и b

Иначе, если такого не произошло, программа восстановит ~~зна~~ строку S до S_0 и продолжит перебирать a и b

Если после работы программы a и b не найдутся, то просто задаём диапазон „n" побольше в самом начале

1. Создадим массив a[n], где, например,

a[1] = $c_1 = 1^1$ и т.д.

Если у последовательности есть период, то есть такое a[i], что a[i] = a[1]

Итак, составим алгоритм

±

```
for (i=2; i<=n; i++)
if (a[i]==a[1])
{t = i-1;
for (j=1; j<=n; j++)
{ if (a[j]!=a[j+t])
{ t=n;
break; } flag=0; }
~~else flag = 1;~~
```

перебираем все a[i] находим равное и предполагаемый период

Внутри цикла проверяем является ли это периодом для остальных элементов

Если программа обнаруживает, что это не период для каких-то a[i] и a[i+t], то цикл завершается, t становится максимальным (= n) ...

...для того, чтобы заново попытаться его найти ~~если~~ (при этом значение флажка зануляется)

```
~~else~~ else flag = 1; }
if (flag == 1)
cout << t
```

Если же после завершения работы программы значение флажка будет равно 1, значит число t действительно является периодом последовательности и мы выводим его на экран

2. Заведём соответствующие массивы каждому ряду a[n1], b[n2], c[n3]

Отсортируем первый ряд

```
for (i = 1; i <= n1; i++)
for (j = n1; j >= 1; j--)
if (a[j] < a[j - 1])
swap (a[j]; a[j - 1])
```

поменять местами элементы

Для каждого элемента начиная с последнего сравниваем с соседним и он перемещается если необходимо. Таким образом каждый элемент либо постепенно „всплывает“, либо „падает“

Отсортируем 2 ~~ой~~ ряд

```
for (i = 1; i <= n2; i++)
for (j = n2; j >= 1; j--)
if (b[j] > b[j - 1])
swap (b[j], b[j - 1])
```

Сортируется также, но по убыванию

Известно, что a[n1] = b[1], т.к. a[n1] максимальный элемент в ряду a, а b[1] - максимальный в ряду b, то далее если a[n1] как либо изменится, он останется максимумом и для 1 ряда и для 2 после сортировки

~~Запомним значение a[1] : amin = a[1];~~

```
for (i = 1; i <= n3; i++)
for (j = n3; j >= 1; j--)
if (c[j] < c[j - 1])
swap (c[j]; c[j - 1])
~~if (c[n3] > amin~~
```

Отсортируем ряд c по возрастанию

Затем, нужно проверить является ли элемент c[n3] (наибольший в ряду c) минимальным для ряда а, так как он находится на стыке

Запомним значение c[n_3] (или a[1] (одно и то-же число))

```
min = c[n3]
```

Применим сортировку для ряда а ещё раз для проверки

```
for (i = 1; i <= n1; i++)
for (j = n1; j >= 1; j--)
if (a[j] < a[j - 1])
swap(a[j]; a[j - 1])
if (a[1] != min)
cout << "Ошибка";
```

Если элемент a[1] не равен min, это значит, что он передвинулся

То есть, для ряда а после сортировки двух других рядов ~~ниже~~ в ~~стыко~~ стыках с этими рядами не будут находится минимальн~~ый~~ое и максимальн~~ый~~ое числа (по крайней мере одновременно)

±

3 Калькулятор ~~так~~ так или иначе - это программа, в работе которой используются типы данных

Вещественный тип данных ограничивается в получении определённого количества знаков ~~после~~ после запятой

То есть в той или иной программе на том или ином языке программирования вещественный тип данных не может дать любое (бесконечно большое) количество знаков в дробной части

Например, ~~в какой~~ пусть можно получить только 5 знаков после запятой

~~Если его много раз~~ ~~вещ~~ Если вещественное число много раз делить на число больше единицы, то в целой части будет 0, а цифры после запятой будут также становиться нулями. Таким образом при ограничении в 5 знаков (условно) после запятой многократным делением можно получить, например, число 0,00000001 и оно округлится до числа $0,\underbrace{00000}_{5\text{ знаков}}$, то есть до 0

Олимпиада школьников «Надежда энергетики»

ИГЭУ

Место проведения

ИН70-75

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Параскун

ИМЯ София

ОТЧЕСТВО Дмитриевна

Дата рождения 16.08.2001

Класс: 11

Предмет информатика

Этап: заключительный

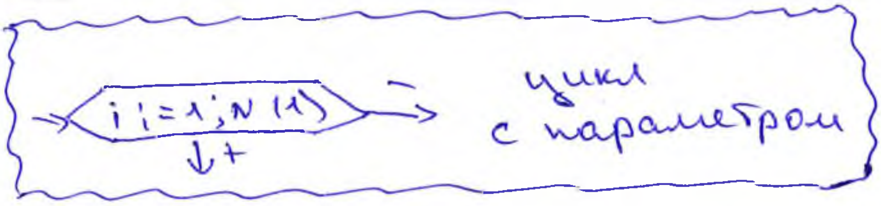
Работа выполнена на 4 **листах**

Дата выполнения работы: 14.02.2019
(число, месяц, год)

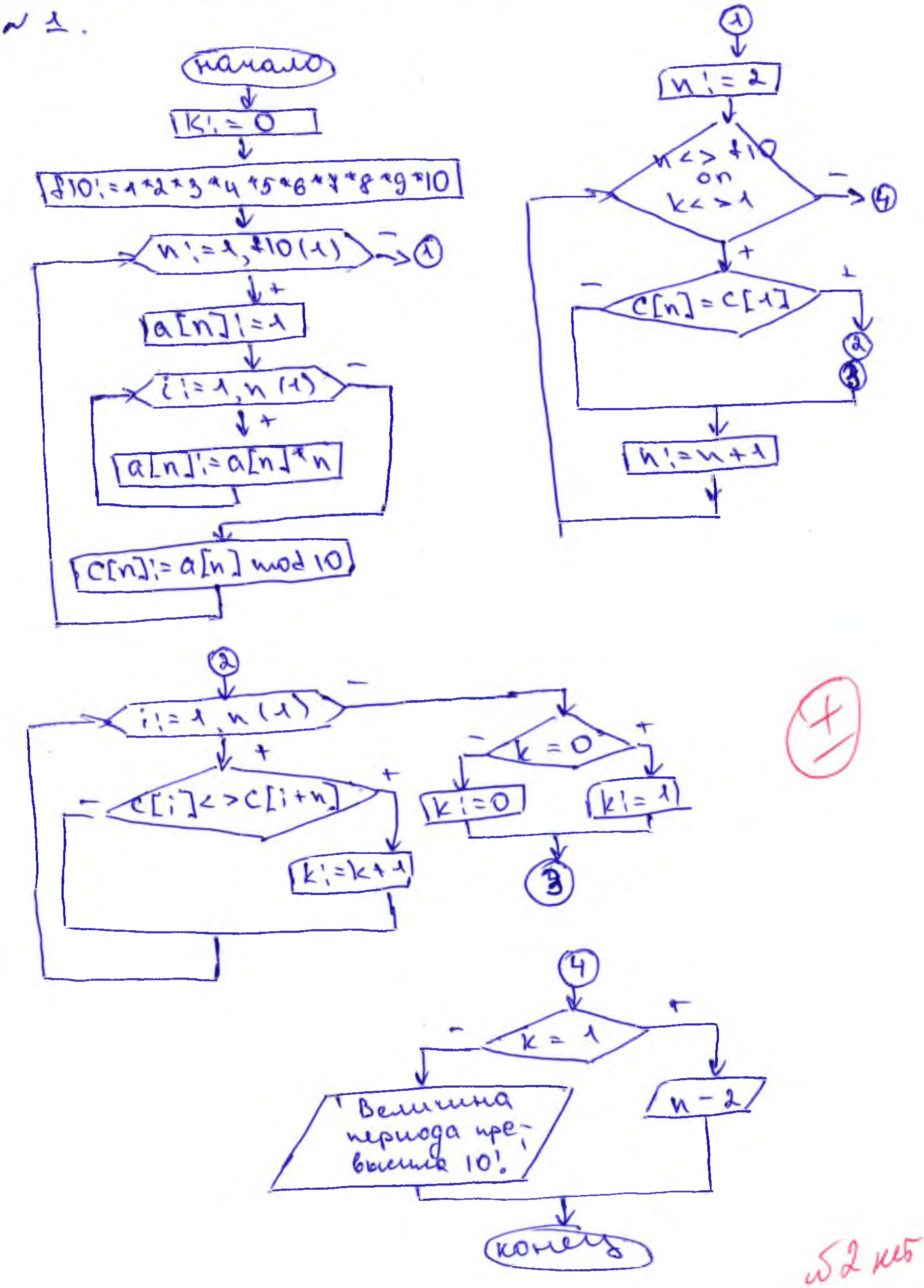
Подпись участника олимпиады: Пар

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



№ 1.



±

ω2 кб

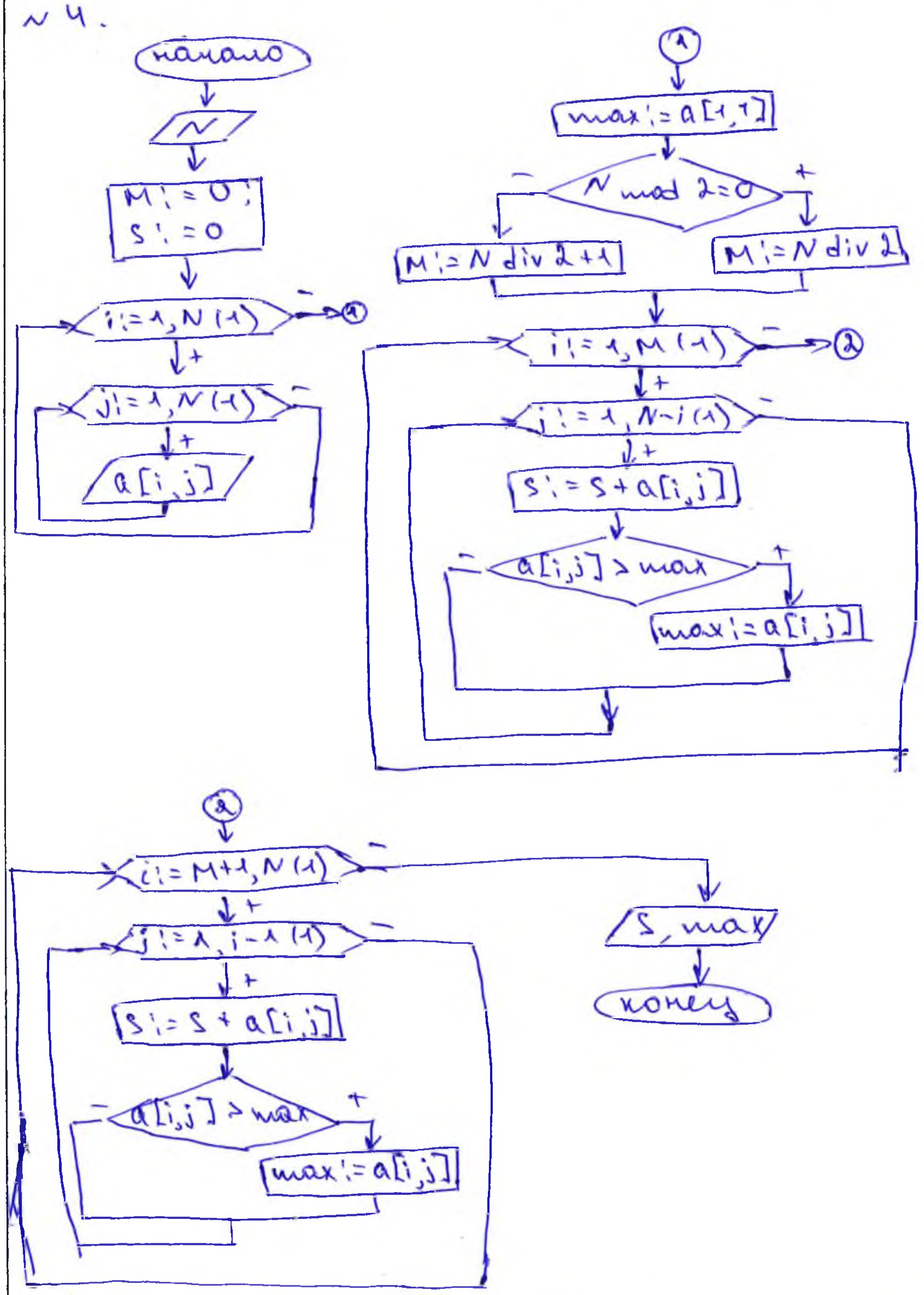
Олимпиада школьников «Надежда энергетики»

Вариант: 73 111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ЙН 70-75

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№ 4.



±

№ 3.

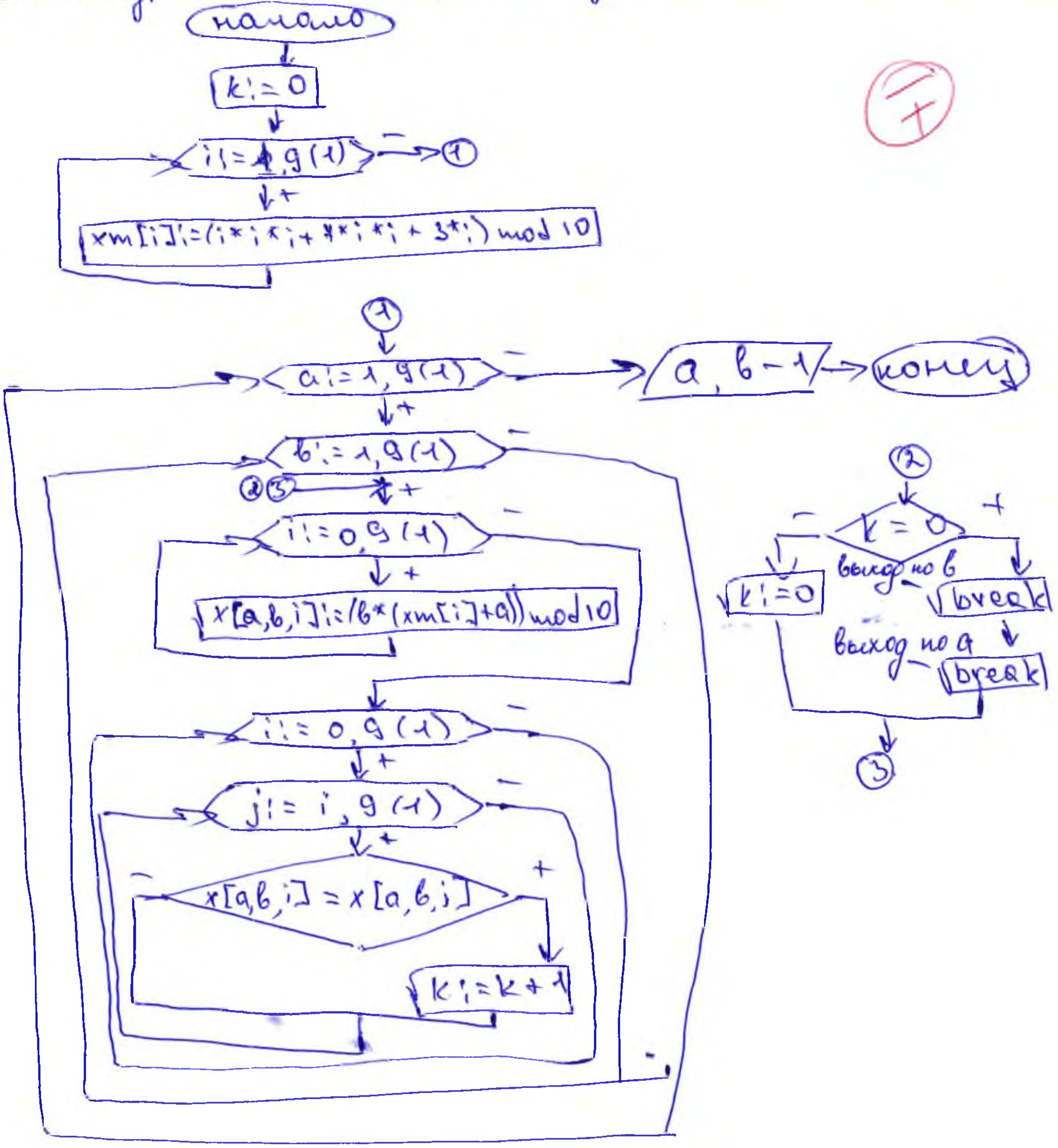
Возможно, получаемый результат был настолько мал, что калькулятор округлил его до 0. На компьютере же это произошло после большего числа повторений, так как разрешающие возможности памяти больше, чем у калькулятора, а следовательно он может хранить более маленькие числа, а значит и округлять незначительно малые числа до 0 он будет позже.

±

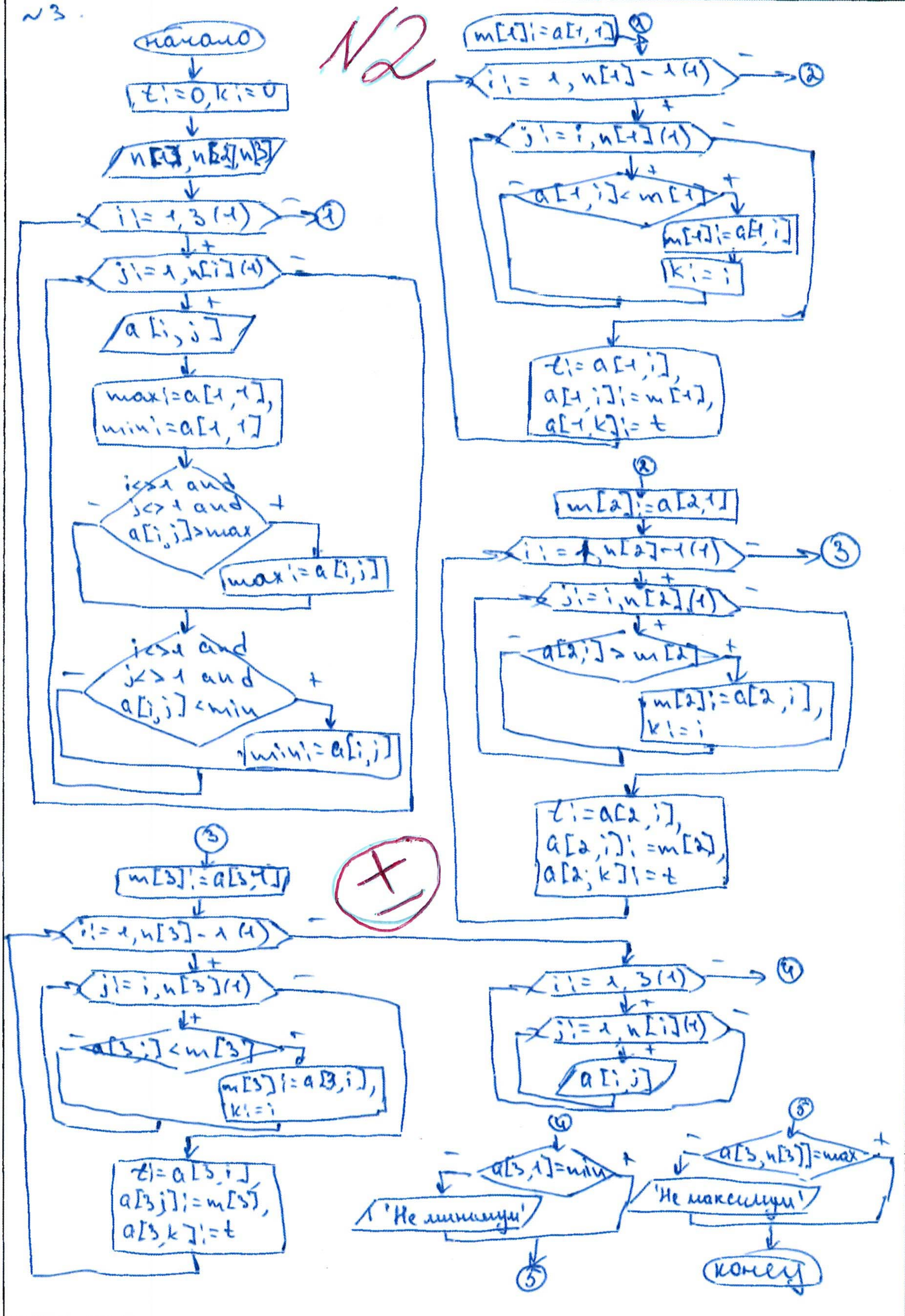
ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№5.

Остаток от деления многочлена F(x) на 10 является последняя цифра полученного числа. Последнюю цифру произведения чисел можно получить при перемножении последних цифр множителей. Тогда можно принять значения чисел a и b за однозначные числа (от 0 до 9), но так как мы знаем, что это натуральные числа, то от 1 до 9.



ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



Олимпиада школьников «Надежда энергетики»

№ группы	Место проведения
	Москва

АИ 32-12

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 93101

ФАМИЛИЯ Пахмутова

ИМЯ Арина

ОТЧЕСТВО Алексеевна

Дата рождения 10.10.2002

Класс: 10

Предмет информатика

Этап: заключительный

Работа выполнена на 4 **листах**

Дата выполнения работы: 17.02.19
(число, месяц, год)

Подпись участника олимпиады: АПах

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, общее количество листов, на которых выполнена работа и дату выполнения работы.

Пр-ма! (+)

```
1. {Free Pascal}
program KeyCombination;
const
        RIGHT = 10;
        LEFI = 10;
        KEY = 99;

var
        x, y, z : ShortInt;
begin
        for x := LEFT to RIGHT do
                for y := LEFT to RIGHT do
                        for z := LEFT to RIGHT do
                                if 3 * Sqr (x) - Sqr (y) - 7 * z = KEY then
                                        WriteLn ('x = ␣', x, 'y = ', y, 'z = ', z)


end
2.
program Triangle;
type
        TArr = array [1..1000, ~~1..1000~~] of Integer;
procedure InitArray (var a: TArr; n : Integer);
var     i : Integer;

begin
        for i := 1 to n do
                Read (a[i])
end;
procedure Swap (var a, b : Integer);
var
        t : Integer;
begin
        t := a;
        a := b;
        b := t
end;
procedure Sort1 (var a : TArr; n : Integer);
var
        i, j : Integer;
begin
        for i := 1 to n-1 do
                for j := 2 to n do
                        if a[i] > a[j] then
                                Swap (a[i], a[j])
end;
procedure Sort2 (var a : TArr; n : Integer);
var
        i, j : Integer;
begin
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
        for i := 1 to n-1 do
               for j := 2 to n do
                      if a[i]<a[j] then
                            Swap (a[i], a[j])
end;
procedure PrintArray (const a : TArr; n : Integer);
var
      i,j : Integer;
begin
      for i := 1 to n do
            Write (a[i],'␣');
      WriteLn
end;
var
      a1; a2; a3 : TArr;
      n, m, z : Integer;
begin
      Read (n, m, z);
      InitArray (a1, n);
      InitArray (a2, m);
      InitArray (a3, z);
      Sort1 (a1, n);
      Sort2 (a2, m);
      if a1[n] = a2[1] then begin
              Sort1 (a3, z);
              if (a2[m] = a3[1]) and (a3[z] = a1[1]) then begin
                          PrintArray (a1, n);
                          PrintArray (a2, m);
                          PrintArray (a3, z)
              end
              else WriteLn ('На стыках между рядами не оказался минимум и максимум')
      end
      else
           WriteLn ('На стыках между рядами не оказался минимум и максимум')
end.
```

програм.

∓

3.

```
program Del;
var
     a, b : Double;
begin
     Read (a, b);
     while a>0 do begen
          a := a/b;
     WriteLn (a)
     end
end.
```

Числа, над которыми Сережа может производить ~~определенные~~ арифметические операции лежат в определенном диапазоне (В моем примере эти числа имеют вещественный 4-байтный тип Double и имеют точность до 14-15 разрядов после запятой). Соответственно, когда ~~число~~ при делении на число в разряды, не равные нулю, вышли за пределы допустимой калькулятором экспоненты, Сережа увидел 0.

(+)

4.

```
program Table;
const
        SIZE = 500;
~~procedure 1~~
type    TArr = array[1..SIZE, 1..SIZE] of Integer;
procedure InitArray (var a : TArr ; n : Integer);
var     i,j : Integer;
begin
        for i := 1 to n do
                for j := 1 to n do
                        Read (a[i,j])

end;
procedure  Sum_Max (const a : TArr ; n : Integer); var max, s : LongInt);
var
        i,j : Integer;
begin   s := 0; max := 0;
        if  n  mod 2 = 0 then
                for i := 1 to n mod 2 do ~~begin~~
                        for j := 1 to n-i do begin
                                s := s + a[i,j];
                                if a[i,j] > max then
                                        max := a[i,j]

                end
        else
                for i := 1 to n mod 2 + 1 do
                        for j := 1 to n-i do begin
                                s := s + a[i,j];
                                if a[i,j] > max then
                                        max := a[i,j]

                        end;
        for i := n - n mod 2 + 1 to n do
                for j := 1 to i do begin
                        s := s + a[i,j];
                        if a[i,j] > max then
                                max := a[i,j]

                end
end;
var
        a : TArr;
        max, s : LongInt;
        n : Integer;
```

Пр-ма!
(+)

Олимпиада школьников «Надежда энергетики»

Вариант: 73101

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ АИ 32-12

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
begin
        Read (n);
        InitArray (a, n);
        Sum_Max (a, n, max, s);
        Write ('Сумма ', s, 'Максимальное число ', max)
end.
5.
  program Code,
  var
        a, b, x, m: Word;
  begin
        Read (x);
        Read (m); { правая граница диапазона чисел a и b}
        for a := 1 to m do
            for b := 1 to m do
                if (Sqr (x)* x + 7 * Sqr (x) + 3 * x + a) * b mod 10 = x then
                    Write (a, '␣', b)

  end.
```

(+)

Алг + программа!

Олимпиада школьников «Надежда энергетики»

СОШ № 20

Место проведения

24 37-82

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 93101

ФАМИЛИЯ Сергеев

ИМЯ Никита

ОТЧЕСТВО Олегович

Дата рождения 31.10.2002

Класс: 10

Предмет Информатика

Этап: заключительный

Работа выполнена на 4 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады: Серг

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

N1

Для нахождения всех отвечающих комбинаций мы можем перебрать все возможные x, y и z с проверкой условия. Т.к числа лежат в диапазоне от 10 до 20, то сложность данного алгоритма будет небольшой $\simeq 10^3$. Все подходящие нам комбинации будем выводить

```
for x in range[10, 20]:
   for y in range[10, 20]:  // перебор по всем x,y,z в диапазоне
     for z in range[10, 20]:   от 10 до 20 включительно
       if 3·(x·x) + y·y - 7·z==99:  // проверка условия
          print(x, y, z) // вывод подходящего нам кода
```

Пр-но! ⊕

N2

Каждый ряд карточек можно представить как отдельный массив чисел, только после изменения в каждом ряде нужно изменять пересекающиеся карточки в других рядах. Для расположения всех карточек в нужном нам порядке мы можем отсортировать массив N3, изображенный на рисунке. После изменяем смежные элементы. Далее отсортируем 1 массив по возрастанию и 2 массив по убыванию со сменой смежных элементов. После сортировки ① и ② на первом и последнем месте ③ могут стоять числа только меньшие в левой части и больше в правой части, так что числа там не будут расположены от минимального к максимальному.

Допустим, что массивы уже заполнены числами.

```
ar_3.sort() // сортируем ③
ar_1[1], ar_2[-1] = ar_3[1], ar_3[-1]
ar_1.sort() // сортируем ①      // замещение элементов
ar_3[1], ar_2[1] = ar_1[1], ar_1[-1] // замещение
ar_2.sort(↓) // сортируем ② по убыванию так, чтобы
                больший элемент находился внизу
```

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
ar_2[1], ~~ar[2~~ ar_2[-1] = ar_1[-1], ar_3[-1] // замещение.
// далее соединим все 3 массива и найдем их общий максимум
 и минимум
Arr = ar_1 + ar_2 + ar_3
Max = max(Arr)
Min = min(Arr)
// затем нужно проверять находится ли минимум на пересечении
 ③ и ①  и максимум на пересечении ③ и ②, если нет,
 вывести сообщение об ошибке
if Max != ar_3[1] or Min != ar_3[1]:
  print("Ошибка").
```

∓

№3

Калькулятор выдал значение 0, т.к у любой ЭВМ есть пределы ~~мощности~~ вычислительной мощности. После деления числа, калькулятор округляет результат до определенного значения. Когда результат становится слишком маленьким числом, то калькулятор округлит его до нуля. Таким образом верена и получим 0 при многократном делении.

?

∓

№4

Найдем закономерность, по которой в таблице расположены белые клетки. Заметим, что от N-ого столбца в левую сторону в белый цвет окрашено на 1 клетку меньше сверху и на 1 клетку меньше снизу. Таким образом мы можем составить заданную таблицу и найти сумму чисел в ее диагональных элементах и максимальный элемент. Допустим у нас есть пустая таблица NxN элементов - A.

```
i = N //столбец
l = 1 //нижняя граница
r = N //верхняя граница
while (r-l+1)>2: //пока можно ставить белые клетки
    for temp in range[l,r]: // в диапазоне от l до r включительно
        A[i][temp] = 'X' //окрашиваем в белый.
l, r = l+1, r-1 //переход на след. шаг
```

```
i = i - 1
// теперь запишем значения во все клетки не являющиеся белыми
for i in range[1;N]:
    for j in range[1;N]:
        if A[i][j] != 'x':
        input(A[i][j]) // ввод числа в закрашенную клетку
// теперь пройдемся по закрашенным клеткам, найдем сумму и максимальный элемент
Sum = 0
Max = 0
for i in range[1;N]:
  for j in range[1;N]:
   if A[i][j] != 'x':
      Sum = Sum + A[i][j] // сумма
      if A[i][j] > Max:
          ~~Max[i]~~ Max = A[i][j]. // поиск максимального
Print(Sum)
Print(Max) // вывод суммы и макс. значения элементов массива
```

Пр-ма!

⊕

если изначально в некоторых ячейках таблицы будут значения, то алгоритм не изменится, т.к. в ~~закрашенные~~ белые клетки мы заходить не будем, а ячейки без числа будем просто пропускать

№5

Для того, чтобы преобразование для каждой цифры а допускало однозначное расшифрование, наша функция $F(x) = b(x^3+2x^2+3x+a) \bmod 10$ для каждой цифры выдавала разное значение. Для этого мы можем просто перебрать значения ~~ох~~ a и b и результаты функции для каждой цифры запишем в структуру данных set, которая не может содержать в себе двух одинаковых элементов. Так, если длина нашего set при определенных a и b будет равна 10, то эти a и b подходят нам для однозначного расшифрования. При a больше 10 его значение по модулю 10 будет таким же, как от 0 до 10, поэтому имеет смысл перебирать a [0;9]

⊕

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

```
S = set() // наш set
b = 1
a = 0
while True :
   for X in range [0; 9] : // проверяем все цифры
      ~~temp~~ temp = $b \cdot (X^3 + 7X^2 + 3X + a) \bmod 10$ // значение функции
   S.insert(temp) ——>      // добавляем в set значение
   if len(S) == 10 : // проверка на кол-во различных значений
      print(a, b)
      break
   else :
      a = a+1
      b = b+1
   if a == 10 : // если a > 9, то его можно обнулить
      a = 0
```

У данного алгоритма есть недостаток. Если ~~так~~ заданных a и b не существует он будет выполняться бесконечно, но тогда и решений данной задачи не существует.

Олимпиада школьников «Надежда энергетики»

ИГЭУ

Место проведения

ЙН70-99

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Сизяков

ИМЯ Иван

ОТЧЕСТВО Голонович

Дата рождения 26.02.2002

Класс: 11

Предмет информатика

Этап: заключительный

Работа выполнена на 7 **листах**

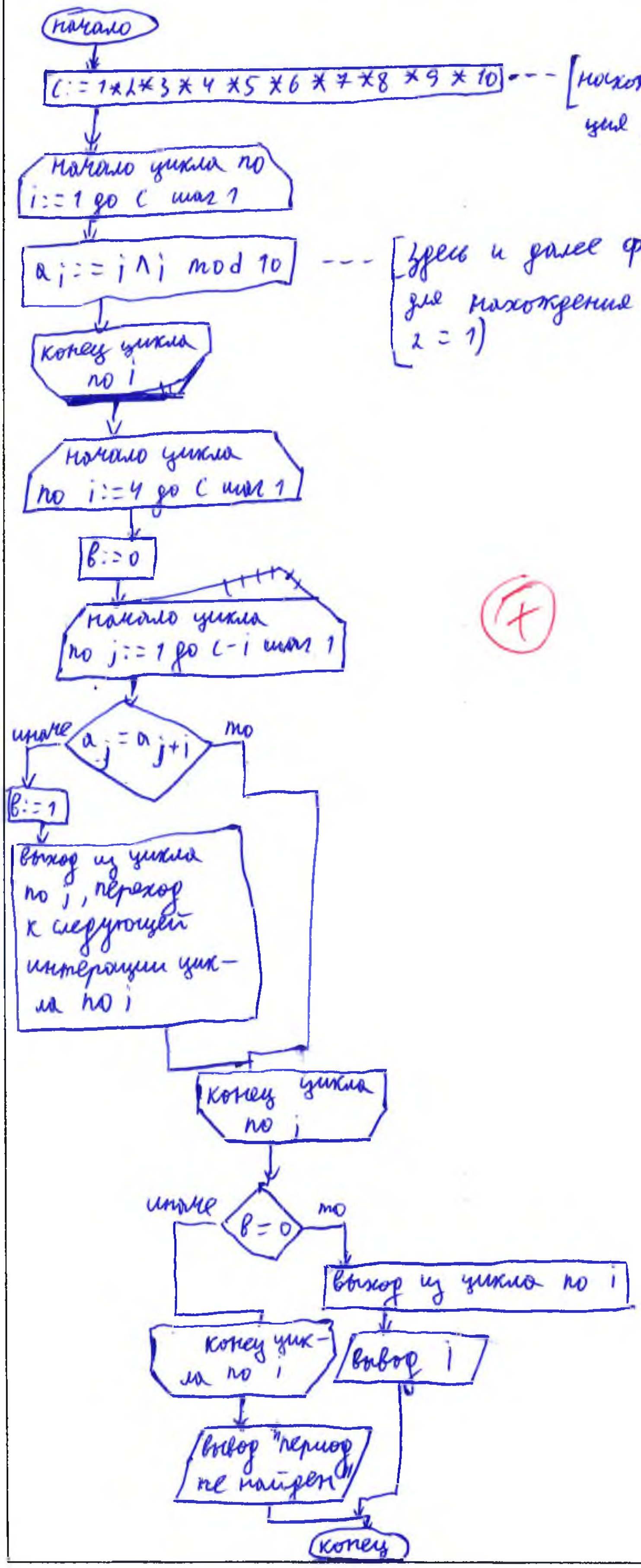
Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады: Сиз

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

N 1



[нахождение 10!, здесь и далее операция умножение обозначается *

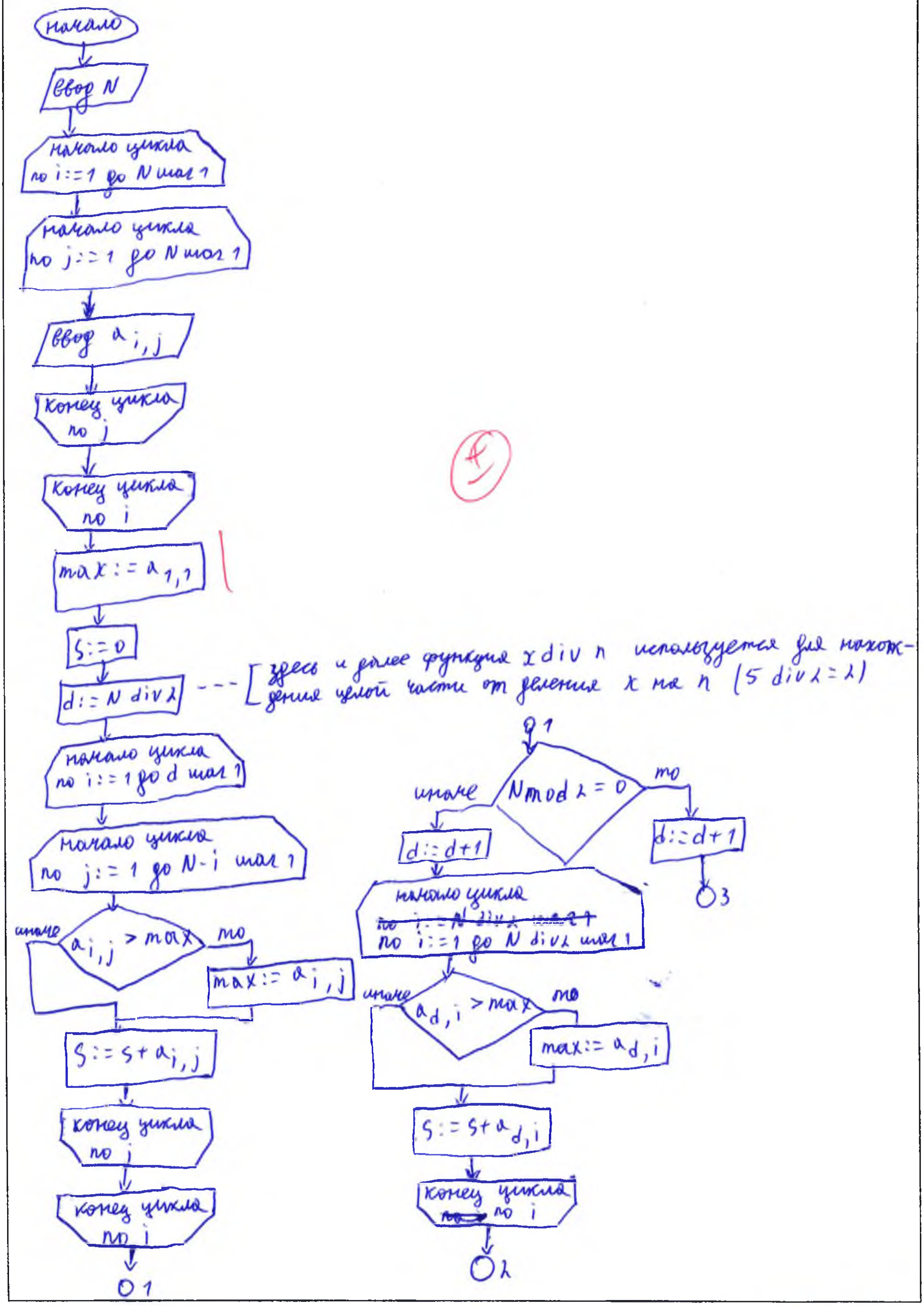
[здесь и далее функция ~~мо~~ x mod n используется для нахождения остатка от деления x на n (5 mod 2 = 1)

+

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

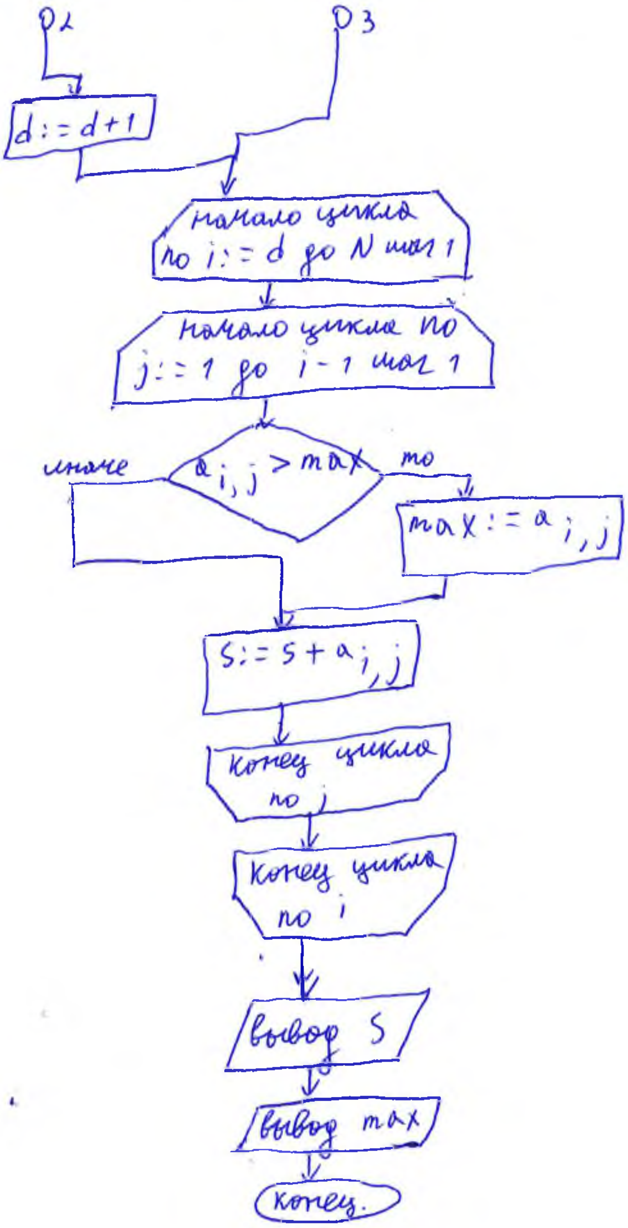
№4

+



[здесь и далее функция x div n используется для нахождения целой части от деления x на n (5 div 2 = 2)

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

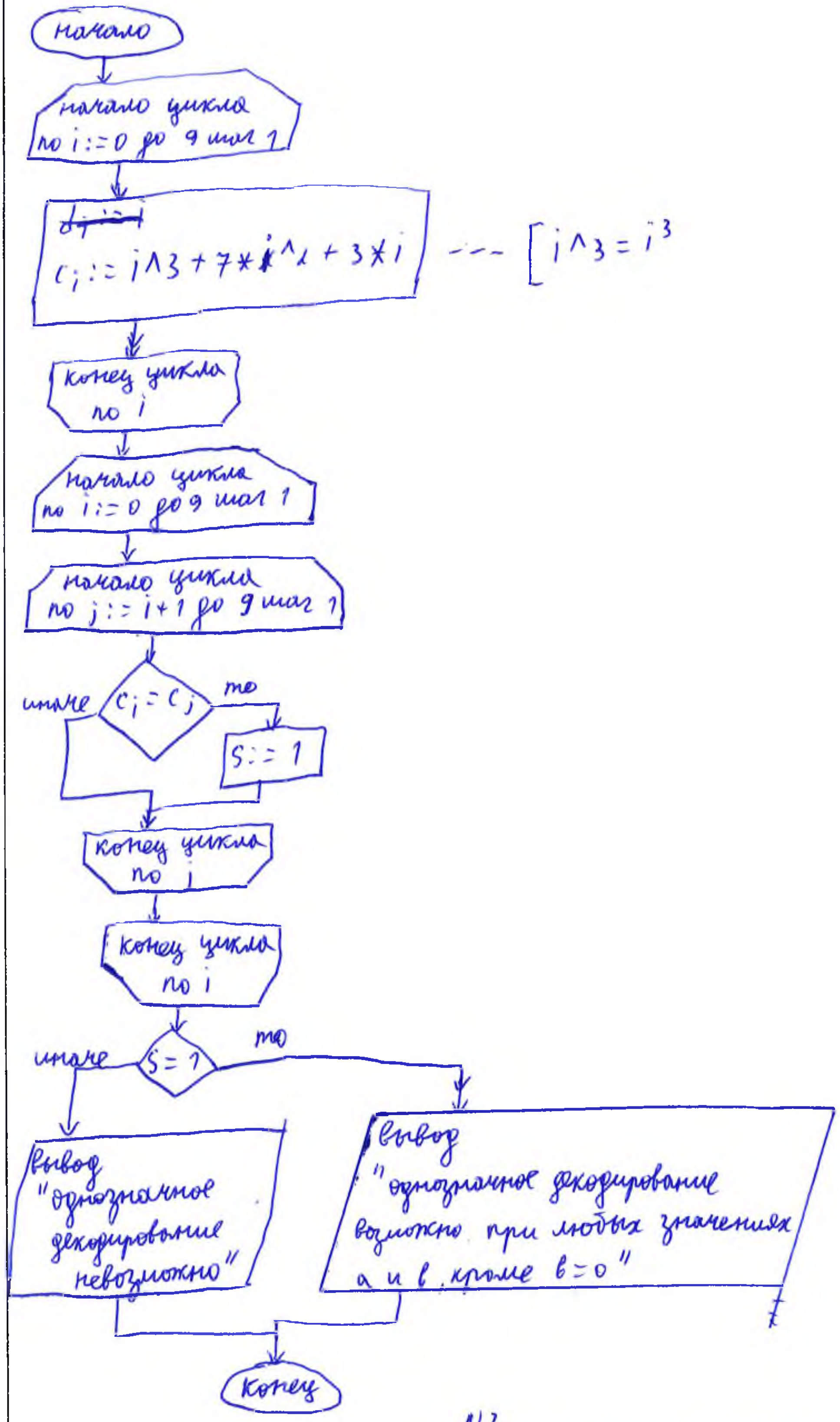


N5

Заметим, что цифр всего 10 (0-9), значит если однозначное декодирование возможно для последовательности 0 1 2 3 4 5 6 7 8 9, то оно возможно для любой другой последовательности различных цифр.

Так же заметим, что если для двух цифр x_1 и x_2 остатки от деления на 10 значения многочлена x^3+7x^2+3x равны, то остатки от деления на 10 значения многочлена $b(x^3+7x^2+3x+a)$ для данных x_1 и x_2 будут равны при любых значениях a и b. Следовательно, ~~либо~~ однозначное декодирование либо возможно при любых значениях a и b (кроме $b=0$), либо ~~невоз~~ невозможно.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



N 3

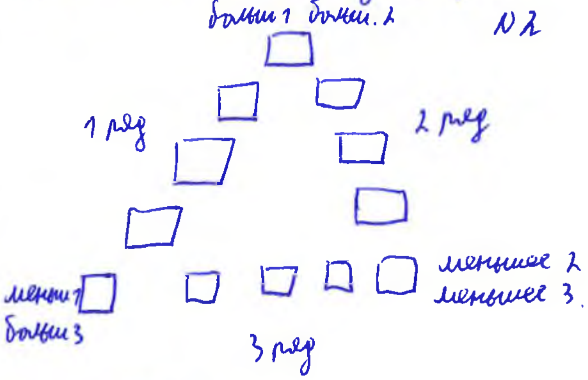
Калькулятор и компьютер могут обрабатывать определённое количество знаков после запятой. Если после деления в результате получается число с большим колечеством знаков после запятой, то происходит округление.

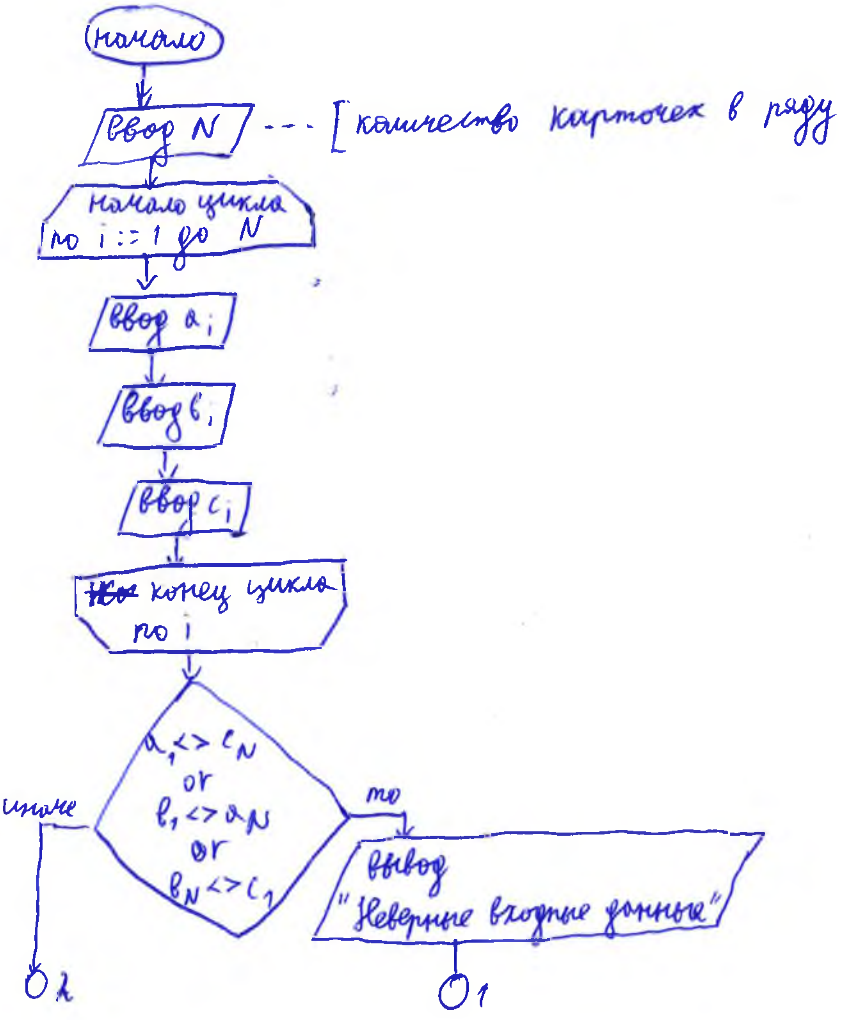
Число 0 будет получено, если после округления все доступные знаки после запятой будут заняты 0 (например, если устройство может вывести 3 знака после запятой, а в результате деления получилось 0,0001, то оно будет округлено до 0).

+

На калькуляторе и компьютере 0 получается после разного числа делений, т.к. компьютер может обрабатывать и выводить больше знаков после ~~зап~~ запятой, чем калькулятор.

№2



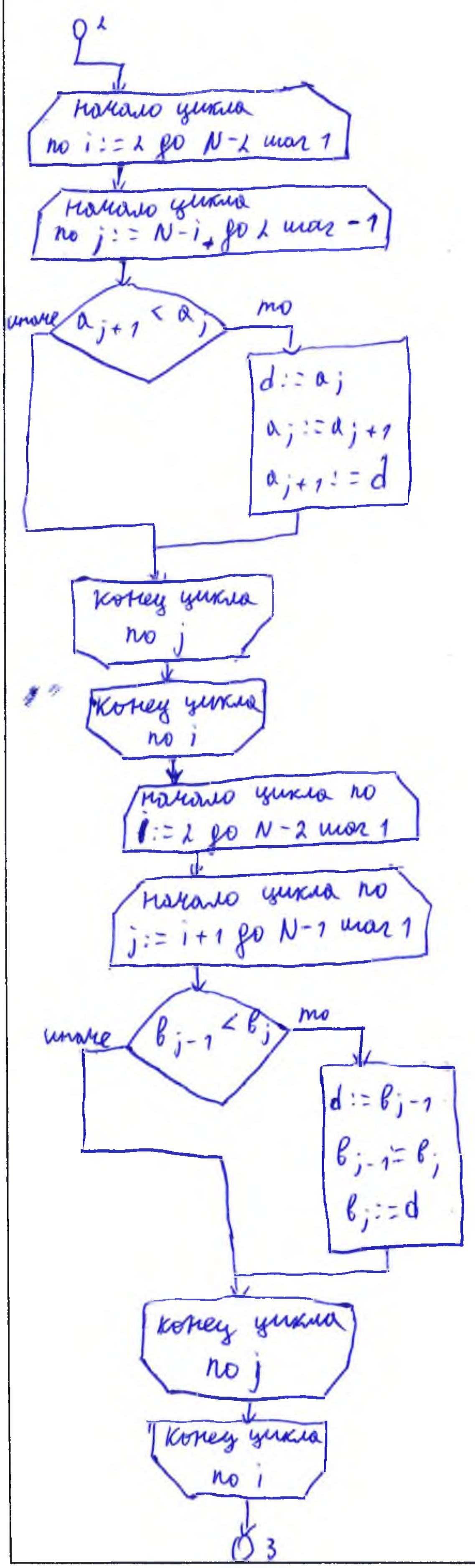


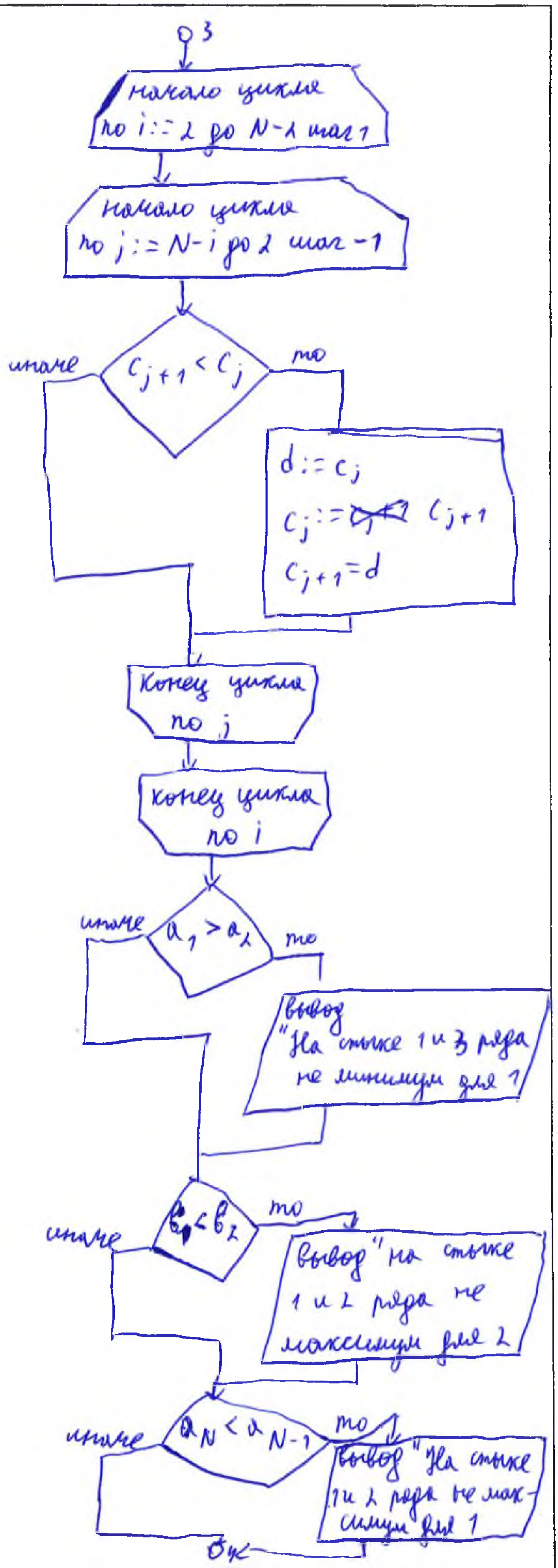
Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ОН70-99

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



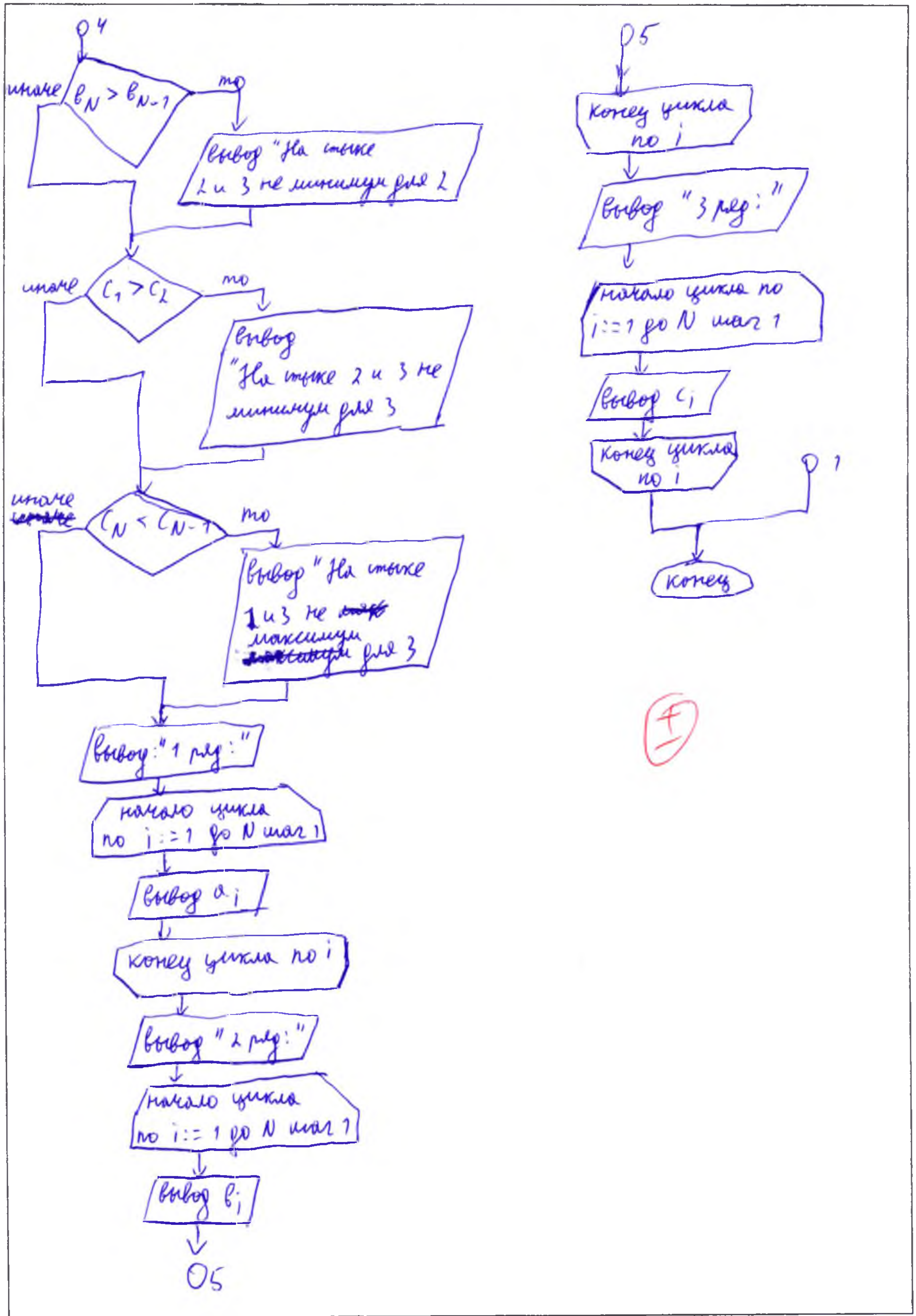


Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ИМ70-99

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



Олимпиада школьников «Надежда энергетики»

МБОУ „СОШ №2"

Место проведения

ИВ 51-30

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 93101

ФАМИЛИЯ Соколов

ИМЯ Никита

ОТЧЕСТВО Витальевич

Дата рождения 19.03.2002

Класс: 10

Предмет Информатика

Этап: Заключительный

Работа выполнена на 2 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады: Соколов

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

N 1.

```
Запускаем цикл $x^{ов}$ от 10 до 20 включит:
  Запускаем цикл $y^{ов}$ от 10 до 20 включит:
    Если $(3x^2 - y^2 \le 239)$, то:
      Запускаем цикл $z^{ов}$ от 10 до 20 включит:
        Если $(3x^2 - y^2 - 7z == 99)$ то
          вывод $x, y, z$
```

(+)

N 3.

Сергей дошёл до положит числа которое меньше мини. допустимого числа и из-за этого калькулятор отбросил числа которое не влезли в память и он мог получить 0

?

Например: калькулятор показывает только 3 ~~знака~~ цифра после запятой. Значит мин положит число равно 0,001. Если разделить это число на 10, то мы должны получить 0,0001, но он отбросит „лишние" цифра и сохранит только 0,000, а это и есть 0.

∓

N 4

Пусть вся табл это двумерный массив a и S изнач равно нулю и maxa равна самому маленькому допустим числу.

```
Запускаем цикл на i от 1 до N включ. {
  Если i ≤ ((N деленное на 2 нацело)+1), то {
    Запускаем цикл на j от 1 до (N-i) включ. {
      S = S + a[i][j]
      Если a[i][j] ≥ maxa, то
        maxa = a[i][j] } }
  Иначе {
    Запускаем цикл на j от 1 до (i-1) включ.
      S = S+a[i][j]
      Если a[i][j] ≥ maxa, то
        maxa = a[i][j] } }
⇐ Выводим S. это и будет сумма всех элем наход в закраш клетках
Выводим maxa - это и есть макс знач элементов.
```

(+)

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

№ 2.

Алгоритм сортировки:

```
Цикл на i от 0 до n (не вкл) (n-кол-во эл. в ряду)
    Цикл на j от 0 до (n-i-1) (не вкл):
        Если n[j] > n[j+1], то
            x = n[j]
            n[j] = n[j+1]
            n[j+1] = x
```

Сортируем по алгоритму выше ряд №1 и получим массив n1 отсорт по возраст.

Отсорт. ряд №2 по тому же алг. и развернем его, получим массив n2 отсорт по убыв.

Отсорт. ряд №3 по тому же алгоритму и получ массив n_3 отсорт по возраст.

+

Если $n_1[0] != n_3[len(n_3)]$ или $n_1[len(n_1)] != n_2[0]$ или $n_2[len(n_2)] != n_3[0]$

то выводим соответствующее сообщение.

Если мы ни разу не вывели соответствующее сообщение, то задача выполнена

№ 5.

Если в ≥ 10, то в можно заменить на (в % 10) и значение не изменится.

Если а ≥ 10, то а можно заменить на (а % 10) и знач не поменяется.

+

```
Цикл в от 0 до 9 (включ)
    Цикл а от 0 до 9 (включ)
        Цикл x от 0 до 9 (включ)
            Если в != 0 и а = 0 и $x^3+7x^2+3x+a==0$
                то выводим все в = числам не оканч на 0
                и а = числам оканч на 0.
```

Олимпиада школьников «Надежда энергетики»

Москва

Место проведения

ZZ 94-28

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73991

ФАМИЛИЯ УГРЮМОВ

ИМЯ МИХАИЛ

ОТЧЕСТВО АНДРЕЕВИЧ

Дата рождения 13.06.2004

Класс: 9

Предмет ИНФОРМАТИКА

Этап: ЗАКЛЮЧИТЕЛЬНЫЙ

Работа выполнена на 3 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Задача 1.

Переберём все варианты и проверим, какие подходят, учитывая, что $y \neq 0$, $x \neq 0$, т.к. yx - произведение. Для ~~всех~~ каждого x от ~~0~~ 1 до 9 будем брать все значения y от 1 до 9. И если после подстановки A будет нацело делиться на yx, выводим x и y.

⊕

Задача 3.

Это произошло потому, что калькулятор выводит числа с ~~какой-то~~ точностью ~~до~~, ~~а у Серёжи получи~~ до 10^{-n}, где n натуральное число, а у Серёжи получилось число, у которого ~~первые~~ n ~~разрядов~~ старших разрядов (а может и больше) были равны нулю. Поэтому калькулятор округлил это число до нуля.

⊕

Задача 5.

⊕

Создадим переменную $i = 1$, которую будем увеличивать на 1 после каждого шага в цикл, причём если $i = 56$, выходим из цикла. На каждом шаге цикла будем проверять следующее:

Если элемент с индексом $i+2$ равен элементу с индексом $i+6$ и не равен элементам с индексами i, $i+1$, $i+3$, $i+4$, $i+5$, то выводим на экран следующее: „слово подъезд может располагаться на месте от", i, " элемента текста до", i+6, " элемента текста".

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Задача 2.

Для первого ряда:

Пусть его длина равна l, тогда для каждого элемента с индексами от 1 до l-1 делать следующее:

Счётчик = 0.

Если текущий элемент больше следующего, то меняем их местами, при этом увеличиваем счётчик на 1.

Повторим это l раз, но если в ~~какой-то~~ после какого-то раза счётчик равен нулю, выходим из цикла.

+

Для второго ряда:

Делаем то же самое, но менять элементы местами будем, если текущий меньше следующего.

Задача 4.

Заметим следующее

число степень	0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8	9
2	0	1	4	9	6	5	6	9	4	1
3	0	1	8	7	4	5	6	3	2	9
4	0	1	6	1	6	5	6	1	6	1
5	0	1	2	3	4	5	6	7	8	9
6	0	1	4	9	6	5	6	9	4	1
7	0	1	8	7	4	5	6	3	2	9
8	0	1	6	1	6	5	6	1	6	1
9	0	1	2	3	4	5	6	7	8	9
⋮										

← последняя цифра числа

всегда в чёт. степени

всегда в ~~чёт~~ нечет степени

Причём чётные числа всегда в чётной степени, а нечётные в нечётной.

Значит последовательность имеет следующий вид: каждый десяток членов может быть только одним из таких:

$1, \boxed{4, 7}, 6, 5, 6, \boxed{3, 6}, 9, 0$

$1, \boxed{6, 3}, 6, 5, 6, \boxed{7, 4}\ 9, 0.$ Причём они будут чередоваться.

Таким образом, эта последовательность состоит из таких кусочков длиной 20:

$\boxed{1, 4, 7, 6, 5, 6, 3, 6, 9, 0,}\boxed{1, 6, 3, 6, 5, 6, 7, 4, 9, 0}$

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

+

Олимпиада школьников «Надежда энергетики»

СОШ №20
Место проведения

ЯУ 34-47
шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Чайкина

ИМЯ Екатерина

ОТЧЕСТВО Юрьевна

Дата рождения 08.03.2001

Класс: 11

Предмет информатика

Этап: заключительный

Работа выполнена на 4 **листах**

Дата выполнения работы: 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

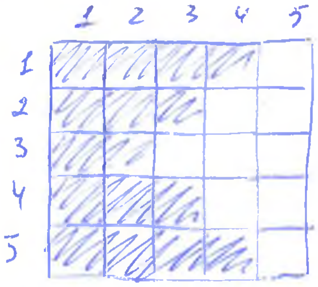
Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ЯУ 34-47

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

4.



Дана таблица размером n×n. Чтобы найти сумму чисел закраш. части сначала нужно определить незакрашенную. Пусть i, j – номера строк и столбцов соответственно.

```
k=0;
for (j=N; j≥n/2+1; j--)
{ for (i = 1+k; i≤ n-k; i++)
  a[i][j]=0;
  k++; }
```

±

В приведенном фрагменте программы на C++ мы перебираем столбцы таблицы начиная с последнего до середины. При этом с уменьшением номера столбца уменьшаются и границы строк (в послед. столбце обнуляются все строки, в предпоследнем все, кроме первого и последнего), т.е. верхние и нижн. границы перебора по строкам с кажд. столбцом уменьшаются на 1 (переменная k в программе – значение отступа). Мы зануляем незакраш. область, т.к. нули не влияют на сумму и макс. значение. Теперь можно пройтись по всем элементам таблицы и если в ячейке есть число и это не нуль, то накапливаем сумму, т.е. прибавляем его значение (условие, где значение не равно нулю проверять не обязательно, как упомянуто выше). Таким образом мы находим сумму всех элементов. Чтобы найти максимальное значение, сначала примем за максимальное элемент с номером (1;1) (верх. лев. угол) и ~~при переборе~~ сравним кажд. элемент таблицы с этим числом. Если очередной элемент больше действ. макс. значения, присваиваем максимальному это значение.

Олимпиада школьников «Надежда энергетики»

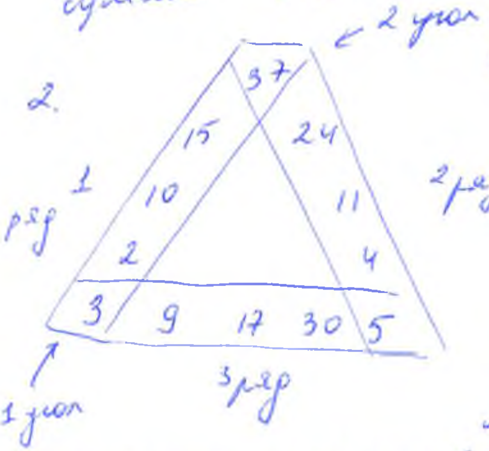
Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ 9У34-47

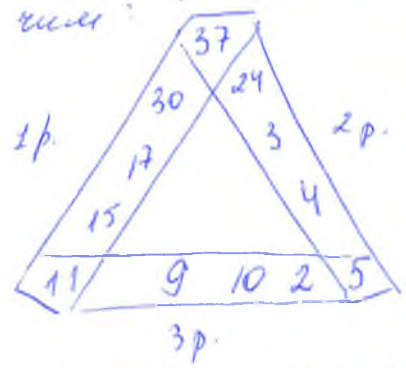
ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Сумма и поиск:
```
max = a[1][1];
      макс. значение элементов
for (i=1; i<=n; i++)
 for (j=1; j<=n; j++)
  if (a[i][j] > max) max = a[i][j];
   S+ = a[i][j];
   сумма элементов
```

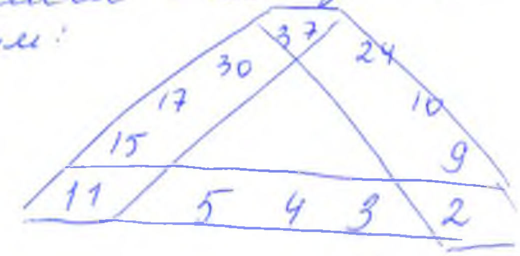
2.



Имеются 3 ряда чисел. Начнем расставлять их с нижнего левого угла. По условию число в нем должно быть одновременно и бо́льшим для 3 ряда, и меньшим для 1 ряда. Тогда логично поставить на его место число со средним значением, но такое, чтобы между ним и числом во 2 углу также имелись числа. Во 2-м углу по условию стоит число, бо́льшее для 2-х рядов, а значит, максимальное. В данном примере бо́льшее число остается во 2-м углу, в первом, например, 11. Тогда, расставив в 1 ряду значение, получим:



Во 2-м ряду числа расположены по убыванию. Число в 3-углу минимальное для 2-х рядов, значит на его место поставим минимальное. Диагональ 3 ряда меньше по значениям, поэтому сначала расположим в нем числа по возрастанию, а оставшиеся — по убыванию во 2-м ряду. Получим:



Fx

Лист 2 из 4

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

5. $F(x) = b \cdot (x^3 + 7x^2 + 3x + a)$

$x^3 + 7x^2 + 3x$ – найдем значения этого трехчлена при $x \in [0; 9]$

7

$x=0 \rightarrow \underline{0}$

$x=1 \rightarrow 1\underline{1}$

$x=2 \rightarrow 4\underline{2}$

$x=3 \rightarrow 9\underline{9}$

$x=4 \rightarrow 18\underline{8}$

$x=5 \rightarrow 31\underline{5}$

$x=6 \rightarrow 58\underline{6}$

$x=7 \rightarrow 70\underline{7}$

$x=8 \rightarrow 98\underline{4}$

$x=9 \rightarrow 132\underline{3}$

Кажд. цифра заменяется остатком от деления $F(x)$
$F(x)$ на 10, т.е. на послед. цифру
Однозначное расшифрование допускается, если
для каждого x ост. от деления $F(x)$ на 10
различен. Из приведенных выше значений
мы видим, что для каждого x остаток от
деления $(x^3 + 7x^2 + 3x)$ на 10 различен, поэтому
сразу можем получить пару чисел $a = 0$,
$b = 1$. Однако a и b по условию – натур. числа
значит $a = 0$ не подходит. Однако мы мо-
жем подобрать такие a, при которых послед.
цифра $(x^3 + 7x^2 + 3x)$ не меняется, т.е. числа, крат-
ные десяти (b при этом также равно 1).
Послед. цифры $(x^3 + 7x^2 + 3x)$ также будут различны,
если прибавить любое a. Если b – четное, то
$2 \cdot 5$ и $2 \cdot 0$ дает одинак. послед. цифру =>
люб. четн. значение b
=> расшифровование не однозначное. Так же и при b
кратном 5 (остатки 0 и 5). ~~четн~~
Получаем, что a может быть любым, а
b – не четным и не кратным 5.

1. 1^1 2^2 3^3 4^4 5^5 6^6 7^7 8^8 9^9 10^{10} ~~-последоват~~-ть

$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$

1 4 7 6 5 6 3 6 9 0 - посл. цифры

При увеличении основания степени на 10 послед. цифра получ. значения не меняется / $(1^1)=(11^1)$, $(2^2)=(12^2)$). (n^n) будем обозначать последнюю цифру n^n.

11^{11}
$\downarrow$
1

$(12^{12})=(2^{12})=((2^2))^6=(4^6)=6$

$(13^{13})=(3^{13})=((3^3)^4\cdot3)=(7^4\cdot3)=3$

предположим, последние цифры повторяются через каждые ~~10~~ n, кратные 10. Как мы видим, при n, отличных на 10, остатки не совпадают, тогда проверим условие для n, отличных на 20.

$(22^{22})=(2^{22})=((2^2)^{11})=(4^{11})=((4^4)^2\cdot4^3)=4$

$(23^{23})=(3^{23})=((3^3)^7\cdot3^2)=(7^7\cdot3^2)=(3\cdot3^2)=7$

$(24^{24})=(4^{24})=((4^4)^6)=(6^6)=6$

$(25^{25})=(5^{25})=5$

$(26^{26})=(6^{26})=((6^6)^4\cdot6^2)=(6\cdot6^2)=6$ ⊕

$(27^{27})=(7^{27})=((7^7)^3\cdot7^6)=(3^3\cdot7^6)=3$

$(28^{28})=(8^{28})=((8^8)^3\cdot8^4)=(6^3\cdot8^4)=6$

$(29^{29})=(9^{29})=((9^9)^3\cdot9^2)=(9^3\cdot9^2)=9$

Мы видим, что для $n\in[1;9]$ остатки такие же, как и для $n\in[21;29]$. Аналогично можно убедиться, что остатки одинаковы для $n\in[11;19]$ и $n\in[31;39]$. Получается, что элем-ты повторяются каждые 20 n.

3. Т.к. при описанном делении получаются нерациональные числа (бесконеч. десятич. дроби), то калькулятор допускает погрешности. При постоянном уменьшении числа в 6 раз число приближается к нулю, остаются тысячные, миллионные доли, которые калькулятор округлит до нуля. В компьютере же вычисление происходит раньше из-за повышенной точности калькулятора или др. вычислительной программы ±

Открытая студенческая олимпиада «Надежда энергетики»

	Москва
№ группы	Место проведения

ДК 50-73

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Шацкий

ИМЯ Ростислав

ОТЧЕСТВО Евгеньевич

Дата рождения 02.07.2001.

Образование: 11

Предмет информатика

Этап: заключительный

Работа выполнена на 5 **листах**

Дата выполнения работы: 17.02.1[illegible]19
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, сведения об образовании, название предмета, общее количество листов, на которых выполнена работа и дату выполнения работы.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

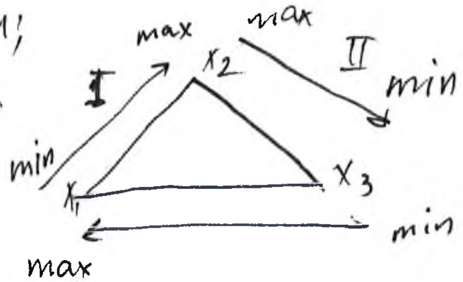
```
№) обозначим переменные
  i, k, a, T : цел;
  f : логич;
  ~~m[1..4000000]~~ m: массив [1..~~4000000~~ 7257600] из цел;
начало
 a:=1;

для i от 1 до 3628800·2 делать
  нц
    для k от 1 до i делать
      a:=(a·i) mod 10;   остаток от деления на 10
    m[i]:=a;
    a:=1;
  кц;
для i от 2 до 3628800 делать                    #4 ⑦
  нц
    для k от 1 до i делать
      если m[k]<>m[i+k]   не равно
        то начало
             f:=ложь;
             выход из цикла;
           конец
        иначе f:=истина;
    T:= i;
    если f=истина то выход из цикла;
  кц;
если f=истина
     то вывод (T)
     иначе вывод ('период либо > 10!, либо не существует);
конец.
```

Открытая студенческая олимпиада «Надежда энергетики»

Вариант: 7311

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ 08К 50-73

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа



```
N2
обозначим переменные
f: логич;
N1, N2, N3, ~~x1, x2, x3~~, i, j, ~~y1, y2, y3~~: цел;
M1, M2, M3: массив [1..20000) из цел;
начало
f:= правда;
ввод (N1, N2, N3);
для i от 1 до N1 делать          } //ввод рядов
  | ввод (M1[i]);
для i от 1 до N2 делать
  | ввод (M2[i]);
для i от 1 до N3 делать
  | ввод (M3[i]);

для i от 1 до N1-1 делать
  | для j от i до N1-1 делать
  |   | если M1[j] > M1[j+1]
  |   |   | то поменять значения (M1[j]; M1[j+1]);
M2[1]:= M1[N1];
для i от ~~1 до 1~~ N2 вниз до 2 делать
  | для j от i вниз до 2 делать
  |   | если M2[j] > M2[j-1]
  |   |   | то поменять значения (M2[j]; M2[j-1]);
если M2[1] >= M1[N1-1]
  | то ~~поменять значения (M2[~~ M1[N1]:= M2[1]
  | иначе f:= ложь;
для i от 1 до N3-1 делать
  | для j от i до N3-1 делать
  |   | если M3[j] > M3[j+1]
  |   |   | то поменять значения (M3[j]; M3[j+1]);
если M3[1] <= M2[N2-1]
  | то M2[N2]:= M3[1]
  | иначе f:= ложь;
если M3[N3] <= M1[1]
  | то M1[1]:= M3[N3]
  | иначе f:= ложь
если f = истина
  | то ~~для i от 1 до N~~ начало
  |        вывод M1;
  |        вывод M2; вывод M3;
```

// N - длина ряда
// производится сортировка карточек и проверяются углы на min и max в определённом ряду

⑦

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

N 2 (продолжение)

```
то начало
  вывод ('I -');
    для i от 1 до N1 делать вывод (M1[i],' ');
  вывод ('II -');
    для i от 1 до N2 делать вывод (M2[i],' ');
  вывод ('III -');
    для i от 1 до N3 делать вывод (M3[i],' ');
  конец
иначе
  вывод ('~~расположение не~~ на стыках рядов не
         оказалось максимум и минимум');
конец.
```

N3

При подсчёте ~~[illegible]~~ иррационального числа происходит округление дробной части до определённого количества ~~знаков~~ цифр в зависимости от количества памяти, выделенной для хранения результата.

Как только последняя допустимая ~~знак~~ цифра дробной части округлится до нуля (~~все нижние~~ и все ~~предыд~~ остальные будут =0), значение результата станет равным "0".

При делении ~~число~~ результат [illegible] постоянно уменьшался, пока все цифры (~~при~~ имеющейся памяти) не стали равны "0". Так он получил "0" на калькуляторе и компьютере.

На компьютере памяти на хранение результата выделяется больше, чем на калькуляторе, поэтому повторять операцию пришлось больше раз.

P.S. Если $b<1$, то результат возрастает, переполняя отведённую память. Думаю, в таком случае был бы выведен идентификатор ошибки, который тоже может быть "0".

+

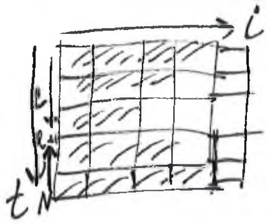
Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ ДК-5073

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

N4



```
Обозначим переменные
 i, t, N, l, S, M: цел;
 K: массив [1..20000; 1..20000] из цел;
начало ввод(N);
S:=0; M:=~~-~~20000;
  для t от 1 до N делать
   для i от 1 до N делать
   | ввод(K[i,t]);
  если N mod 2=0          остаток от деление ~~целочислейн~~
   | то l:=N div 2         целочисленное деление
   | иначе l:=N div 2+1;
  для t от 1 до ~~N~~ l делать
  | для i от 1 до N-t делать
  |   нц
  |   | S:=S+K[i,t];
  |   | если K[i,t]>M то M:=K[i,t];
  |   кц;
  для t от N ~~до~~ вниз до l+1 делать
  | для i от 1 до t-1 делать
  |   нц
  |   | S:=S+K[i,t];
  |   | если K[i,t]>M то M:=K[i,t];
  |   кц
 вывод('сумма=', S, ' макс. значение=', M);
end.
```

+

N5

```
Обозначим переменные
M: массив [0..9] из цел;
a, b, Na, Nb, i, j: цел;
f: логич;
начало                     ~~// границы~~ границы перебора a и b
 ввод(Na, Nb);
 для b от 1 до Nb делать
  для a от 1 до Na делать
  нц
   для i от 0 до 9 делать
   | M[i]:=(b*(i*i*i+7*i*i+3*i+a)) mod 10;     // остаток от деление на
   для i от 0 до 8 делать
   | для j от i+1 до 9 делать
   |  | если M[i]<>M[j]
   |  |  | то f:=истина
   |  |  | иначе ~~f:=ложь~~
   |  |  |  | начало
   |  |  |  | f:=ложь;
   |  |  |  | ~~выход из цикла~~, идти в[1];
   |  |  |  | конец;
                                // проверка, разные ли все члены массива M
```

+

Лист 04 из 05

```
1:
~~кц~~
    если f=истина то выход из цикла;
кц
если f=ложь то вывод('нет значений а и в на данном промежутке
        [1;Na] и [1;Nв]')
            иначе вывод('a=',a,' b=',в);
конец.
```

~~В этой програм~~ Эта программа записывает в массив M значения функции при перебираемых а и в при значениях x от 0 до 9 (знач записывается в ячейку с номером x). После провереется, есть ли среди элементов массива M одинаковые цифры, если их нет, циклы прекращаются и выводятся значения а и в.

Команда идти в [1]; перемещает действие в "1:", нужна, чтобы выйти из двойного цикла.

Олимпиада школьников «Надежда энергетики»

Москва.

Место проведения

ДК 50-96

шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Шукшов

ИМЯ Андрей

ОТЧЕСТВО Игоревич

Дата рождения 19.09.2001. **Класс:** 11

Предмет информатика **Этап:** заключительный

Работа выполнена на 5 **листах** **Дата выполнения работы:** 17.02.19.
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

Для начала посчитаем последовательность до 2·10! и запишем все в строку.

```
string s;
For (int i=0; i < 2·10!+1; i++) {
    int x := i;
    For (int j = 0; j < i; j++) {
        x = (x*j) %10;
    }
    s.push_back(x);
```

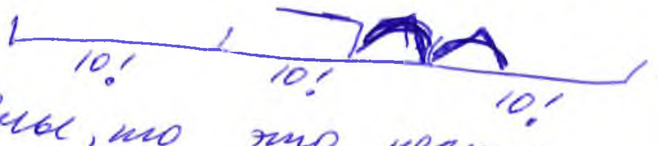
⊕

В итоге мы имеем строку длины 2·10!

Теперь будем просто перебирать суффиксы от длины 1 до 10! (ln), ~~чтоб~~ просто будем, поставлять ~~его~~ после суффикса взятый суф. его.

Получается, просто $\underbrace{123}\,\underbrace{123}_{\text{след}}$ берем строку длины ~~подряд~~ , которую мы взяли. и сравниваем, если они равны, то посчитаем. след ln. элементов. и если они тоже совпадают, то это период.

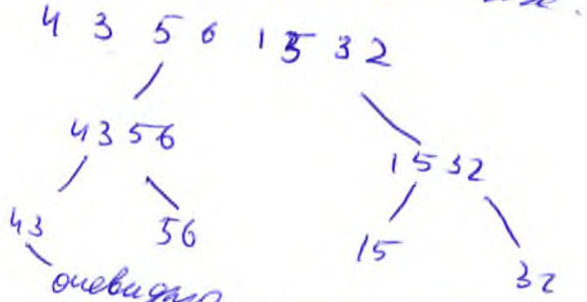
На самом деле в итоге получается, что нам просто ~~до~~ надо взять и посчитать 3·10! значений подряд.



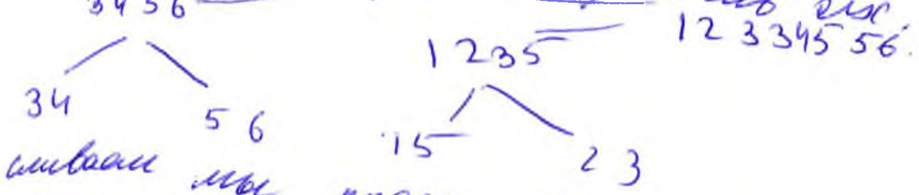
— если все 3 части равны, то это период

№2.

Для начала нам потребуется просто научиться сортировать. Мы можем сортировать за O(n²) просто используя сортировку пузырьком, но если мы хотим быстрее (O(n·logn)), то напишем сортировку слиянием. Это сортировка работает по принципу „разделяй и властвуй". Мы будем брать делить массив на две части и сортировать каждую по отдельности, а после сливать их.



очевидно, что мы можем отсортировать 2 элемента просто сравнив их.



сливаем мы просто при помощи двух указателей указывающих на текущей элемент в одном из двух блоков. Берем меньший элемент из двух и двигаем указатель, если один из массив закончился, то просто допишем второй после, потому все элементы, которые ~~больше~~ в нем находятся гарантированно больше.

ВНИМАНИЕ! Проверяется только то, что записано с этой стороны листа в рамке справа

Продолжение №2.

Вернемся к нашей задаче.
Сначала просто отсортируем 1 и 2 массив, как нас просят по условию (1 - по возрастанию, 2 - по убыванию) (в сортировке использовать только оператор сравнения)
в итоге мы получим 3 массива.
2 из них отсортированы.
теперь ~~[illegible]~~ массивы $(\overset{1}{a}, \overset{2}{в}, \overset{3}{c})$

+

$c[0] = a[0]$; $c[n-1] = в[n-1]$.

теперь просто пройдем и проверим, что ~~если~~ $c[0]$ - min в массиве, а $c[n-1]$ - max.
Если это неверно, то выведем соот сообщение.
Иначе просто отсортируем массив c.

№3.

При деление двух целочисленных переменные на разных устройствах кол-во делений не может отличаться

1 => происходило деление переменных типо double или Float. Очевидно, что при деление в компьютере возникает погрешность. В данном случае она зависит от кол-во. чисел после запятой, сколько мы храним, потому что чем меньше чисел после мы храним, тем больше мы округляем и отбрасываем.

~~[illegible]~~ А ноль мы получили, потому что погрешность и округление присутствует в обоих случаях, просто во 2 она меньше из-за ~~хранение~~ большей точности.

+

Олимпиада школьников «Надежда энергетики»

ШИФР НЕ ЗАПОЛНЯТЬ! ⇨ AK5096

№4.

$\frac{n+1}{2}$ – элементов ~~n-1, n-2, n-3~~

создадим ~~динамический~~ массив

(~~Например в C++ это будет~~ vector<int>)

~~vector<int> a;~~

```
~~for~~    int pos=0;  int a[n];
                      a[0]=n-1;
   for (int i=0; i<(n+1)/2; i++) {
         a[i]=a[i-1]-1;
         pos++;
   }
                              if (n%2==0) {
                                a[pos]=a[pos-1];
                                pos++; }
```

мы получим. $\overset{n=6}{\{5, 4, 3\}}$ = a и pos = 3.

Пр-но!

~~while~~

```
for (int i=pos; i<n; i++) {
    a[i] = a[i-1]+1;
}
```

±

после всего этого a будет включать в себя

n=6 a = {5, 4, 3, 3, 4, 5}

остальное просто пройти по матрице

m – матрица.

```
int sum=0;
int maxim=-1e9;
for (int i=0; i<n; i++) {
   for (int j=0; j<a[i]; j++){
      sum+=m[i][j];
      maxim=max(maxim, m[i][j]);
   }
}.
cout << sum << " " << maxim;
```

Лист 04 из 05

№5.

~~F(x) %10 10.~~

~~$F(x) = 6 / (x^3 + 7x^2 + 3x + a)$~~

~~Очевидно, что при шифрование по модулю 10, шанс коллизии очень высок. и какие тут a и в не брать мы имеем только (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)~~

Просто ~~бе~~ переберем значение a и b допустим до 1000.

7х

Alг ≠ прог.

```
For (int a=0; a<=1000; a++){.
  For(int b=0; b<=1000, b++){
    set<int> st; ( используем множество ).
   For (int i=0; i<10; i++){.
    // перебираем цифры.
      st.insert( F(x) %10 10);.
   }.
   IF(st.size() ==10) {
     cout << a << "_" << b << endl;
    }
  }
 }
```

Set - это множество, оно хранит только уникальной элементы

```
St.insert(1);
(St = {1};)
St.insert(2);
(St = {1, 2};)
St.insert(1);
(St = {1; 2})
```

Получаемся, если размер set. равен 10, то для каждой цифры мы получаем уникальное значение.

Олимпиада школьников «Надежда энергетики»

ВФ МЭИ
Место проведения

LH48-12
шифр

← **Не заполнять**
Заполняется ответственным работником

Вариант № 73111

ФАМИЛИЯ Юрова
ИМЯ Полина
ОТЧЕСТВО Михайловна

Дата рождения 16.05.2001 **Класс:** 11

Предмет информатика **Этап:** заключительной

Работа выполнена на 7 **листах** **Дата выполнения работы:** 17.02.2019
(число, месяц, год)

Подпись участника олимпиады:

Впишите свою фамилию имя и отчество печатными буквами, дату рождения, класс, название предмета, этапа Олимпиады, общее количество листов, на которых выполнена работа и дату выполнения работы.

№1.

Последняя цифра степени числа зависит только от последней цифры числа и не зависит от остальных разрядов.

Докажем это. Пусть число $n = 10a + b$, где $b \in [0; 9]$.

Тогда b — последняя цифра числа.

$n^2 = (10a+b)^2 = 100a^2 + 20ab + b^2 = 10(10a^2+2ab) + b^2$.

Слагаемое $10(10a^2+2ab)$ не влияет на последнюю цифру, т.к. оно делится на 10 ⇒ она зависит только от последней цифры и ~~равна~~ такая же, как в квадрате этой цифры. Для других степеней тоже последняя цифра — b^n.

Тогда нам достаточно рассмотреть цифры от 0 до 9 и их последние цифры в степенях.

n	1	2	3	4	5	6	7	8	9	0
n^1	1	2	3	4	5	6	7	8	9	0
n^2	1	4	9	6	5	6	9	4	1	0
n^3	1	8	7	4	5	6	3	2	9	0
n^4	1	6	1	6	5	6	1	6	1	0
n^5	1	2	3	4	5	6	7	8	9	0

Заметим, что:

1) Для 4 чисел (1, 5, 6, 0) последняя цифра равна этому числу всегда.
2) Для 2 чисел-квадратов (4 и 9) последняя цифра чередуется с периодом 2.
3) Для остальных 4 чисел (2, 3, 7, 8) — последняя цифра чередуется с периодом 4.

(потому что, как мы видим, если степень 2, например, снова закончилась на 2, то дальше пойдёт снова 4, 8, 6, 2, 4, 8, 6 и т.д.)

Поэтому, для определения последней цифры числа n^n — необходимо заполнить двумерный массив 10×4.

10 — вариантов последней цифры.

4 — максимум разных цифр в степенях.

Пронумеруем 10 столбцов таблицы от 0 до 9, а 4 строки таблицы — от 1 до 4 (только вместо 4 поставим 0).

Получаем таблицу:

степень \ цифра	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	0	1	4	9	6	5	6	9	4	1
2	0	1	8	7	4	5	6	3	2	9
3	0	1	6	1	6	5	6	1	6	1

Пусть нам дано число n. Теперь мы можем найти последнюю цифру n^n.

Обозначим массив, как a[x;y].
x – строки, y – столбцы.
Тогда, для нахождения последней цифры n^n необходимо:

1) Найти последнюю цифру числа n (т.к. мы доказали, что последняя цифра в степени зависит только от неё).
y:= n mod 10.

2) Найти остаток деления степени n^n на 4.
(т.к. последние цифры в степени повторяется 4 раза)
X: = n mod 4.

Значит, исходной цифрой будет являться элемент массива a[x,y].
Можно записать функцию. (с заполненным заранее массивом)

```
  function posl (n: integer): 0..9;
begin readln (n);
    x: = n mod 4;
    y: = n mod 10;
    posl (n) := a[x, y];
end;
```

④

Заметим, что если последовательность повторяется с определённым периодом, то каждые T чисел должно повторяться значение x (n mod 4) и y (т.е. n mod 10).
Очевидно, что периодом будет являться число 4·10=40 (т.е. каждые 40 чисел последовательность будет повторяться, т.к. будут совпадать одновременно остатки по 4 и на 10)
Осталось выяснить, является ли T=40 наименьшим периодом.
Заведём новый массив c: array [1..40] of byte;
Заполним его последними цифрами чисел, применяя функцию posl

```
for i:= 1 to 40 do
~~c[i]:=~~
  c[i] : = posl( i);
```

Затем пройдём по нему всеми делителями числа 40 и посмотрим, будет ли последовательность повторяться чаще чем 40 раз. d – делитель числа. ~~flag:=0~~;

```
 ~~for d:= 1 to 20 do~~
 flag: =0; d:= 1;
 while (flag=0) and (d <=20) do flag:=1
   if 40 mod d =0 then begin i:=1 repeat until (i=40 div d) (flag=0) or [illegible]
                ~~for i:=1 to 40 div d do~~
                if C[i]: = c[d*i + i] then i:=i+1
 d:=d+1;                                else flag: = 0;
```

В итоге, если переменная flag получает значение 1, т.е. находится период (d) меньше 40, то выводим это d. Иначе он равен 40.

Олимпиада школьников «Надежда энергетики»

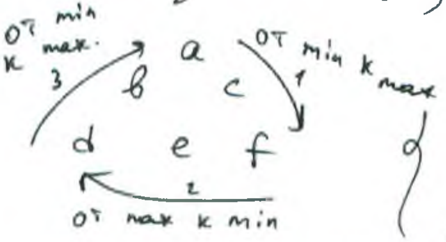
Вариант: 73111.

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ LH48-12

№2.

Рассмотрим решение, когда в каждом ряду по 3 карточки.

Обозначим их буквами.
В 1 ряду – от min к max,
во 2 ряду – от max к min,
в 3 ряду – от min к max



Запишем условия для каждого из рядов.

$$\begin{cases} (1) & f > c > a \\ (2) & d < e < f \\ (3) & d < b < a. \end{cases}$$

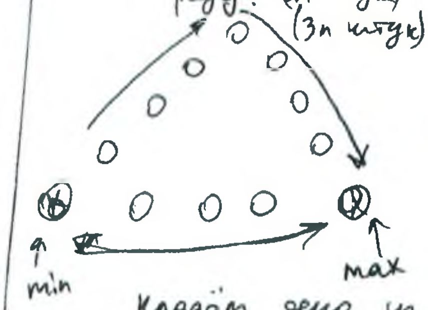
Заметим, что (1) ряд можно объединить с (3)

Тогда: $\begin{cases} f > c > a > b > d \\ f > e > d \end{cases}$

Видим, что f – максимальное число, а d – минимальное. Остальные числа нужно расположить в порядке возрастания. При этом, заметим, что:

1) Если минимальных или максимальных элементов больше 1, то расставить так числа нельзя, т.к. не будет выполняться условие, что f больше всех чисел и d меньше всех чисел
2) Если среди средних чисел (не равных min и max) есть 2 одинаковых числа – ставим одно во (2) ряду, а другое – в 1 или третьем ряду
3) Если среди средних чисел больше 2 одинаковых, то так расположить нельзя.

Выводим алгоритм для любого количества карточек в ряду: ~~3n штук~~) (3n штук)



1) Если есть max и min число – ставим max в правый угол, а min в левый угол.
2) Если среди остальных чисел больше 2-х одинаковых, решения нет
3) Если 2 одинаковых – делим все числа на две группы, в одну кладём одно из этих чисел, в другую – другое

⊖

Если карточек 3n, то в каждом ряду – n+1 штук. ⇒ в одной группе будет n–1, а в другой 2n–1 карточек.

4) Располагаем n-1 в нижнем ряду, а 2n-1 через 2 верхних ряда от min к max по направлению.

№3.

Это происходит из-за округление чисел при делении на калькуляторе и на компьютере

Числа в памяти устройств хранятся, как определённое количество памяти и не могут занимать информационный объём больше указанного. Поэтому, при делении определённая часть числа (если оно не делится нацело), после округления числа отбрасывается. Рано или поздно, если повторять деление большое количество раз, число округлится до 0, т.к. остаток станет слишком мал, чтобы попадать в обозначенный ~~объём памяти~~ и символов. Разное количество повторений деления наблюдается, потому что у устройств разное количество памяти и символов, предназначенных для вывода числа на экран (у компьютера их больше, поэтому и деление повторялось большее кол-во раз. Покажем это на примере. Допустим, у калькулятора максимум 6 символов на экране, а у компьютера — 10. Возьмём какое-то число (например, 123456) и будем делить его на 10.

Калькулятор:	Компьютер
123456	123456
1,2345,6	12345,6
...	...
1,23456	1,23456.

До числа 1,23456 всё идёт одинаково, но затем начинаются отличия, 1,23456 : 10 = 0,123456. Компьютер выведет это число, а калькулятор нет (т.к. в записи уже 7 символов, ⇒ калькулятор отбросит часть символов и округлит

0,12346	0,123456
~~0,1234~~ 0,01235	0,0123456
0,00124	0,0123456
0,00012	0,00123456
0,00001	0,00012346
0,00000 = 0	0,00001235.

Видим что если у устройства больше памяти и оно может работать с более точными числами, то после деления 0 должен получиться позже, чем когда предназначен меньший объём памяти.

Олимпиада школьников «Надежда энергетики»

Вариант: 73111

ШИФР, НЕ ЗАПОЛНЯТЬ! ⇨ LH48-12

№4.

Дана таблица $n \times n$. Обозначим столбцы за x, а строки за y. Найдём, каким условием можно задать запрещенную часть (или не запрещенную).

Пересечение в таблице получается при наложении двух диагоналей одна на другую. Напишем уравнение для не запрещенной части.

Уравнение запрещенной диагонали:

$$x + y = n + 1.$$

Тогда всё, что ниже её, будет задаваться как $x+y > n+1$, а всё, что выше, как $x+y < n+1$.

↖ $x+y > n+1$

Вторая диагональ.

Уравнение диагонали: $y = x$.

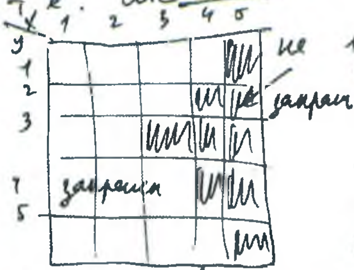
Тогда, всё что выше неё: $x > y$,

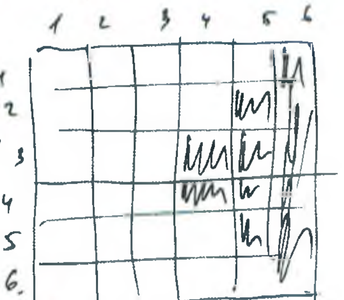
а всё, что ниже её: $x < y$

(это верно и для чётных, и для нечётных n)

Незапрещенная часть задаётся объединением этих двух условий, т.е. системой:

$$\begin{cases} x+y \geqslant n+1 \\ x \geqslant y. \end{cases}$$





Запишем логически

Пусть $(x+y \geqslant n+1) = a$, а $(x \geqslant y) = b$

Тогда система – это их умножение логически (и). $a \cdot b$.

Нам нужна запрещенная часть, т.е. отрицание этих условий

$\overline{a \cdot b} = \overline{a} + \overline{b}$. ⇒ следовательно для запрещенной части будет совокупность их отрицаний

$$\begin{cases} x+y \geqslant n+1 \\ x \geqslant y. \end{cases}$$

↑ не запрещенная

(в программировании and)

$$\left[\begin{array}{l} x+y < n+1 \\ x < y \end{array}\right.$$

↑ запрещенная

(в программировании or).

При написании программы, система, т.е. логическое умножение „и" будет означать and, а совокупность, т.е. логическое сложение „или" – как or. Напишем программу на языке Pascal.

Числа целые, не обязательно натуральные ⇒ переменную max мы не можем в начале взять за 0, т.к ~~от~~ максимальное число может быть и отрицательным.
⇒ берём max за первое встреченное нами число в запрещенной части таблицы. (с помощью переменной flag)
Обозначим a[x,y] - данный двумерный массив (таблицу)

```
flag := 0;
S := 0;
for x := 1 to n do          //по столбцам таблицы
   for y := 1 to n do       //по строкам таблицы.
   ~~begin~~ if (x+y<n+1) or (x<y) then ~~begin~~ begin
               if (flag=0) then begin
                                   max := a[x,y];
                                   flag := 1;
                                end
               else ~~het~~ if (max < a[x,y]) then
                                   max := a[x,y];
               S := S+a[x,y];
             end;
```

(+)

Т.е. если данный элемент - первое встречнное нами число, то мы берём переменную max, равную нему. Если это уже не первый элемент) и значение переменной max у нас есть, то мы сравниваем max и новый элемент.
Прибавляем в сумму (S) каждое встреченное число из запрещенной (т.е. подходящей под условие $\left[\begin{array}{l} x+y<n+1 \\ x<y \end{array}\right.$ части).
В конце выводим S - сумму переменных и max - наибольшее значение

№ 5.

~~$F(x) = b(x^3+7x^2+3x+a)$ mod~~ 10.

Пусть при шифровании цифра x заменилась на цифру y. Тогда

(±)

$y = F(x) \bmod 10 = b(x^3+7x^2+3x+a) \bmod 10.$

Нам нужно, чтобы преобразование можно было однозначно расшифровать, т.е. чтобы для каждого полученного y существовало только одно значение x, из которого можно получить этот y.
Всего 10 различных цифр. x - старая цифра, y - новая цифра после преобразования Значит, никакие 2 и более цифры x не должны шифроваться одинаково ⇒ ~~[illegible]~~ в числе y каждая из цифр от 0 до 9 тоже должна быть по 1 разу

Пусть $g(x) = x^3 + 7x^2 + x$.

Тогда $y = в(g+a) \bmod 10 = (вg + вa) \bmod 10$.

~~При~~ При этом $(вg + вa) \bmod 10 = вg \bmod 10 + вa \bmod 10$.

Докажем это. Пусть $вg = 10k+n$, а $вa = 10p+m$

Тогда. $(10k+n + 10p+m) \bmod 10 = (10(k+p) + n+m) \bmod 10 =$
$= (n+m) \bmod 10$.

При этом, т.к. ва – будет прибавляться к каждому числу при шифровании, мы можем это не рассматривать, т.к. оно прибавляется к каждому, и ва = const.

Рассмотрим все $g(x) = x^3 + 7x^2 + x$ для всех цифр.

x	0	1	2	3	4	5	6	7	8	9
g(x)	0	1①	4②	9⑨	18⑧	31⑤	48⑥	70⑦	98④	132③

Заметим, что последние цифры у чисел g(x) тоже все разные. Значит, т.к необходимое нам число зависит от в·g(x) mod 10, нужно найти такое в, чтобы все в·g(x) имели разные последние цифры.

Это так же будет зависеть только от последней цифры в.

Найдём такие цифры от 0 до 9, чтобы их перемножение со ~~в~~ всеми цифрами имели разную последнюю цифру.

1) Все чётные не подходят (0,2,4,6,8), т.к при умножении не может получиться нечётное
не чётное в конце не цифра

2) 5 – не подходит, т.к. при умножении на любое даёт 5 или 0.

Рассмотрим 1, 3, 7 и 9.

Число: последняя цифра при умножении	0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8	9
3	0	3	6	9	2	5	8	1	4	7
7	0	7	4	1	8	5	2	9	6	3
9	0	9	8	7	6	5	4	3	2	1

Эти 4 числа подходят ⇒ в получается как любое число с одной из этих цифр на конце.

Ответ: в = 10k+1, 10k+3, 10k+7, 10k+9 ~~(k∈Z~~) (k ∈ N)
а – любое