1. **Задача A (100 баллов) «Ограждение»**

Задача «Ограждение» заключается в том, что бы определить длину оградительной ленты необходимо определить суммарную длину прямых участков и суммарную длину скругленных участков. Так как лента достаточно туго натянута, то прямые участки располагаются на прямой, касательной к соседним, бочкам. Длина этого участка будет равна расстоянию между центрами соседних бочек.

Суммарная длина скругленных участков, так как контур замкнутый, будет равна длине окружности, с радиусом равным радиусу бочки.

2**. Задача B (100 баллов) «Ход конем»**

Задача заключается в том, что бы перебрать в необходимом порядке все клетки игрового поля и проверить возможность попадания в нее шахматного коня из исходной клетки.

Переведём координаты заданной клетки из шахматной нотации в математическую. То есть игровое поле представим в виде двумерного массива. После этого переберем в необходимом порядке все поля шахматной доски и выводим те из них, в которые может попасть конь с заданной клетки.

Конь может попасть в данную клетку если вертикальная координата отличается на два а горизонтальная координата отличается на один или если вертикальная координата отличается на один а горизонтальная координата отличается на два.

3**. Задача C (100 баллов) «Тетрис 3D»**

Для решения данной задачи необходимо найти максимальную высоту стопки пластинок, получившихся при падении.

Так как нас интерисует только один параметр – высота получаемой стопки, то для работы целесообразно хранить максимальную высоту в ячейках двумерного массива, соответствующего размеру игрового поля. Последовательно перебирая падающие пластинки необходимо определить максимальную высоту под пластинкой m_i и записать ячейки, перекрываемые пластинкой значения равные (m_i+1). После выполненияи такой операции над всеми пластинками необходимо найти максимальное заначение элементов двумерного массива. Это и будет искомая величина.

4**. Задача D (100 баллов) «Параллелепипед»**

Задача «Параллелепипед» заключалась в том, чтобы по известным длинам двенадцати спичек проверить, можно ли из них склеить каркас параллелепипеда.

Следует вспомнить, что параллелепипед – это фигура, основанием которой служит параллелограмм – четырехугольник, у которого стороны попарно параллельны. У параллелепипеда есть три размерности в 12 ребрах – длина, ширина и высота. Следовательно, необходимо ответить на вопрос, есть ли среди входных данных ровно три группы по четыре одинаковых числа.

Чтобы понять это, необходимо отсортировать все числа и убедиться, что получилось три группы по четыре одинаковых числа (одинаковых по крайней мере внутри каждой группы). Если это так, выводить yes, иначе – no

5**. Задача E (100 баллов) «Числа Шолмса»**

Всего существует 6 перестановок из aa, bb и cc.

Каждую перестановку проверяем на соответствие реальной дате.

Сохраняем все и убираем одинаковые.

6**. Задача F (100 баллов) «Страница дела №X»**

Необходимо последовательно перебирать место разбиения последовательности с учетом:

1. Если длина второй части уже короче, чем длина первой, то это разбиение уже не подходит.
2. Если длины частей равны, то нужно просто сравнить 2 длинных числа
3. Если вторая часть “длиннее” и не начинается с 0 – то это разбиение нам подходит
4. Если длина строки 1, то ответ всегда 0
5. Если второе число начинается на 0, то его считать не надо

7**. Задача G (100 баллов) «Гонки на электромобилях»**

Одна машина проедет *x* div (*tv*) целых сегментов длинной *tv*, сделает между ними *x* div (*tv*) – 1 зарядок батарей. Если *x* mod (*tv*) не 0, то надо проехать ещё, а для этого зарядить батарею. Таким образом, число зарядок: ceil(*x* / (*tv*)) – 1. (Для c++),

Остальное время едет со скоростью *v* .

Время для одной машины *x* / *v* + (ceil(*x* / (*tv*)) – 1) * *t*.

Необходимо сравнить время, за которое машины достигнут финиша

8**. Задача H (100 баллов) «Венероходик»**

Эта задача является классикой динамического программирования. Рассмотреть все возможные маршруты и просчитать их невозможно. Но можно свести задачу к аналогичной. Пусть нам известен “максимальный” путь для всех клеток, кроме правой нижней (функция F(X, Y)). Все нужные маршруты проходят через одну из клеток, смежных с этим углом (их всего две). Максимальный же маршрут проходит через ту клетку из двух, для которой значение функции F больше. Остается только правильно выполнить отсечение:

```
Function F(x,y:integer):longint;
begin
 if B[x, y] = –1 then
     if F(x-1, y) > F(x, y - 1)
     then B[x, y] := F(x - 1, y) + A[x, y]
     else B[x, y] := F(x, y - 1) + A[x, y];
 F := B[x, y]
end;
```

Теперь необходимо подумать о граничных условиях. Логически правильнее было бы просчитать нашу функцию для левой и верхней границы. Это делается легко, так как для этих клеток существует только один маршрут (непосредственно вдоль границы). Но еще проще ввести в рассмотрение фиктивные нулевые строку и столбец и присвоить им нулевые значения. Действительно, эти клетки, в принципе, недостижимы, поэтому максимальная сумма равна нулю.

Итеративное заполнение массива также довольно просто. После введения граничных условий (любых из рассмотренных выше) дальнейшее заполнение осуществляется двойным циклом:

```
for i:=1 to N do
for j:=1 to N do
if B[i - 1, j] > B[i, j - 1]
then B[i, j] := B[i - 1, j] + A[i, j]
else B[i, j] := B[i, j - 1] + A[i, j];
```

9**. Задача I (100 баллов) «Олимпиада 60»**

Для того чтобы разместить журналистов в наименьшем количестве номеров, необходимо сначала расселить по 3-х местным номерам. Количество 3-хместных номеров в данном случае n div 3. Если n mod 3 = 0 , тогда ответ n/3. При n mod 3 = 1, понятно, что 4-х корреспондентов нужно будет расселить в 2 двухместных номера, следовательно ответ (n-4) /3 + 2. В случае n mod 3 = 2, то эти 2 участника займут один номер и тогда ответ n div 3 + 1. Таким образом количество двухместных номеров всегда не превышает 2.

10**. Задача J (100 баллов) «Перестнойка»**

По условию задачи одно существующее ребро (=дорога) удаляется, и добавляется другое в место, где его раньше ребра не было. Нужно посчитать число вариантов такой реформы. Также по условиям задачи известно, что реформа должна приводить к связному графу. Изначально количество групп несвязных между собой городов-вершин может быть любым.

Очевидно, что при числе компонент связности более двух сделать в итоге связный граф добавлением одного ребра у нас никак не получится, ведь так мы свяжем только две группы между собой, а их — больше.

Следовательно, в этом простейшем случае ответ — ноль вариантов.

В случае, если имеется две группы, то есть компонент связности две, то количество вариантов рассчитывается следующим образом.

Сначала нужно найти, какие ребра являются «мостами» — ведь при удалении этих ребер все сводится к первому, рассмотренному выше варианту. Добавить их же назад нельзя, так как по условиям задачи добавить можно только там, где ребер не было. Поскольку в итоге необходимо получить связный граф, добавленное ребро должно соединять вершины из разных компонент связности – делать из несвязного графа связный. Две группы из N и M вершин можно связать между собой NxM способами. Следовательно, число реформ у нас получится NxMxC, где C = число ребер без учета тех из них, что являются мостами.

Третий случай, когда имеется изначально связный граф (это самый сложный случай). В этом случае компонента связности одна, но некоторые ребра могут являться мостами. В итоге число вариантов реформ является суммой двух слагаемых: первое – число вариантов удаления ребер-не-мостов, второе – число вариантов удаления ребер-мостов.

Общее количество ребер можно посчитать по числу вершин: (n * (n-1))/2 (где n = число вершин). Количество ребер, которые можно добавить, вычисляется вычитанием из общего количества числа имеющихся ребер. Так получается первое слагаемое – число ребер, не являющихся мостами, помноженное на число ребер, которые можно добавить.

При удалении «моста» необходимо соединить ребром две получившиеся компоненты связности, развязанные группы вершин. Количество способов это сделать равно произведению вершин справа от моста на количество вершин слева от моста, уменьшенное на единицу, так как сам мост обратно поставить нельзя по условию задачи.