Краткие методические рекомендации по решению задач, 9-11 класс

При разработке задач для основных туров муниципального этапа олимпиады по информатике Региональная предметно-методическая комиссия исходила из того, что все задачи должны быть оригинальными, разнообразными по тематике и не требовать для своего решения специальных знаний.

При определении уровня сложности авторы задач исходили из того, что комплект задач должен содержать как задачи, доступные многим участникам муниципального этапа, так и задачи, позволяющие проявить себя наиболее сильным участником. В этой связи количество задач на основном туре равно пяти, и как минимум две задачи из них такой сложности, что большинство школьников должны их решать полностью, включая и школьников более младших классов. Более того, задачи являются многоуровневыми и предполагают наличие как полных, так и частичных решений, что также будет способствовать тому, что ни одна задача из предложенного комплекта не останется без внимания участников, а сильным участникам позволит продемонстрировать все свои лучшие качества.

Задачи пробного тура предназначены для того, чтобы участники смогли проверить все особенности компьютерной техники и программного обеспечения на своем рабочем месте. Из четырех задач пробного тура задачи W и X являются совсем простыми. Задачи Y и Z предназначены для того, чтобы участники могли лучше познакомиться с системой получения информации о результатах окончательной проверки на примере достаточно сложных задач.

Из пяти задач основного тура **задача 1 «Сенсор»** является самой простой и ориентирована на широкий круг участников.

Задача 2 «Шахматная доска»

Для решения этой задачи нам понадобится таблица, для каждой клетки которой будем помнить, вырезана она или нет. Типа boolean будет вполне достаточно для этого, вначале все клетки пометим false, а по мере вырезания будем записывать в соответствующую ячейку таблицы true.

Таким образом, все вырезанные клетки будут помечены true. Периметр вырезанной фигуры будет складываться из единичных сторон клеток, которые разделяют вырезанную и не вырезанную клетки. Для решения достаточно пройти по всей таблице и, если ячейка вырезана, посмотреть четырех ее “соседей”. Если соседняя ячейка не вырезана, то мы нашли границу вырезанной фигуры и прибавляем к счетчику единицу. Все остальные

случаи соседства (вырезанная с вырезанной и не вырезанная с не вырезанной) никак не влияют на искомый периметр.

Единственная проблема при таком решении – обработка ячеек на границе доски. Можно написать отдельную проверку, но существует более простой способ – барьерный метод. Он состоит в том, что вокруг таблицы создадим “рамку” единичной ширины из не вырезанных ячеек. Несложно заметить, что наличие такой рамки никак не влияет на ответ, но избавляет нас от необходимости отдельно обрабатывать границы доски. Естественно, искать вырезанные клетки на рамке не нужно.

Задача 3 «Без пробелов»

Блок из *k*-значных *p*-ичных чисел содержит числа из диапазона $[p^{k-1}; p^k - 1]$, следовательно, его длина $L_k = k \times (p - 1) \times p^{k-1}$. Определим, в каком блоке и на какой позиции находится искомая цифра, последовательно вычитая L_1, L_2, L_3, … из заданного *n*, пока *n* не станет отрицательным, и затем, вернувшись в предыдущий блок:

```
k := 1; stepen := 1;
while n > 0 do begin
        l := k * stepen * (p - 1);
        dec(n, l);
        inc(k);
        stepen := stepen * p;
end;
inc(n, l); dec(k);
stepen := stepen div p;
```

определим, в каком по счету числе и в каком его разряде находится искомая цифра:

```
nom_chisla := (n + k - 1) div k;
raz := n mod k;
if raz = 0 then inc(raz, k);
```

Найдем значение числа, содержащего искомую цифру, в 10-ичной системе счисления, а затем и саму цифру:

```
if k = 1 then chislo_10 := nom_chisla
        else chislo_10 := stepen + nom_chisla - 1;
for i := 1 to k - raz do
        chislo_10 := chislo_10 div p;
cifra := chislo_10 mod p;
if cifra < 10 then writeln (figure.out, cifra)
```

elsewriteln(*figure.out*, Chr(Ord('A') + *cifra* - 10));

Задача 4 «Кошка и Мышка»

Пусть M_1, M_2, M_3 – точки, соответствующие выходам из норки.

Если треугольник $M_1M_2M_3$ – остроугольный, то кошка *K* должна занять позицию в центре описанного вокруг $M_1M_2M_3$ круга.

Если этот треугольник – тупоугольный (или прямоугольный), то положение кошки – в середине наибольшей стороны $M_1M_2M_3$.

Если точки M_1, M_2, M_3 лежат на одной прямой, то положение кошки будет по-прежнему в середине отрезка наибольшей длины. Во всех случаях положение кошки определяется однозначно.

Задача 5 «Трафареты»

Решение задачи основано на использовании динамического программирования, и основная идея такого решения заключается в следующем. Подсчитаем *m[n]* – количество способов нарисовать непересекающиеся отрезки с вершинами в заданных *n* точках.

Выделим одну из точек, например, первую, и разобьем способы на два непересекающихся класса: те, которые содержат отрезок, выходящий из точки 1, и те, в которых такого отрезка нет. Количество способов первого класса, очевидно, равно *m[n – 1]*. Все способы второго класса содержат некоторый отрезок, выходящий из точки 1. Пусть, это будет отрезок, соединяющий точки с номерами 1 и *k + 1*. По разные стороны от этого отрезка расположено *k - 1* и *n - k - 1* точек. Количество способов провести непересекающиеся отрезки на первой дуге равно *m[k - 1]*; количество таких способов на второй дуге равно *m[n - k - 1]*. По правилу произведения общее число всех таких комбинаций равно *m[k - 1] * m[n - k - 1]*. Суммируя по всем *k* от 1 до *n – 1*, получим окончательную формулу для подсчета чисел m[n]:

$$m[n] = m[n - 1] + \sum_{k=1}^{n-1} m[k-1] * m[n-k-1],$$

с начальным условием m[0] = 1.

Приведем несколько первых значений последовательности *m[n]*:1, 1, 2, 4, 9, 21, 51, 127, …

Замечание. Последовательность m[n] играет важную роль в дискретной математике. Эти числа называются ***числами Моцкина***