

Всероссийская олимпиада школьников по информатике

Вологодская область

II (муниципальный) этап

2016-2017 учебный год

7 - 8 классы

Методические рекомендации по разбору задач

Задача 1. Запись дисков (100 баллов)

Пусть N – число дисков, K – число компьютеров, M – время записи одного диска. Чтобы найти время для записи всех дисков, нужно поделить число дисков на число компьютеров, округлить полученный результат в большую сторону и умножить на время записи одного диска. Основная сложность здесь в том, что отсутствует операция округления вверх – её необходимо реализовать через имеющиеся операции. Это делается так: N / K с округлением вверх можно записать как $(N + K - 1) / K$. Напомним, что операция $'/'$ в формуле означает деление нацело.

Тогда ответ задачи запишется так:

$$(N + K - 1) / K * M$$

Разумеется, это не единственный правильный ответ – участник можно записать правильную формулу и в другом виде. Проверка решения осуществляется подстановкой тестовых наборов значений переменных N , K , M в формулу участника. За каждый верный ответ даётся 10 баллов. Тестовые данные и ответы к ним имеются в материалах к задаче.

Для автоматизации проверки решений этой задачи могут использоваться те же самые программные средства, что и для проверки остальных задач.

Задача 2. Количество чисел (100 баллов)

При $N = 1$ ответ равен 9.

При $N = 2$ первая цифра может быть любой от 1 до 9, вторая – любой от 0 до 9, кроме первой. Итого $9 \cdot 9$ вариантов. К этому ещё нужно добавить ответ для $N = 1$, то есть ответ равен $9 + 9 \cdot 9$.

При $N = 3$ первая цифра может быть любой от 1 до 9, вторая – любой от 0 до 9 кроме первой, третья – любой от 0 до 9 кроме первой и второй. К этому ещё нужно добавить ответ для $N = 2$. То есть ответ равен $9 + 9 \cdot 9 + 9 \cdot 9 \cdot 8$.

Продолжив аналогичные рассуждения, получаем при $N = 4$ ответ $9 + 9 \cdot 9 + 9 \cdot 9 \cdot 8 + 9 \cdot 9 \cdot 8 \cdot 7$, при $N = 5$ ответ $9 + 9 \cdot 9 + 9 \cdot 9 \cdot 8 + 9 \cdot 9 \cdot 8 \cdot 7 + 9 \cdot 9 \cdot 8 \cdot 7 \cdot 6$, и так далее.

Пример решения на Free Pascal:

```

var
  n, i, ans, f: longint;
begin
  read(n);
  if n = 1 then begin
    writeln(9);
    exit;
  end;
  ans := 9;
  f := 9;
  for i := 2 to n do begin
    f := f * (11 - i);
    ans := ans + f;
  end;
  writeln(ans);
end.

```

Задача 3. Родители и дети (100 баллов)

Задача представляет собой изменённую и значительно упрощённую версию задачи "Возраст" для 9-11 классов.

Для начала заметим, что если в семье меньше трёх детей, то искомой даты не существует — можно сразу выводить ответ -1 и завершать программу.

Прочитаем даты рождения родителей и детей в одномерные массивы. В качестве текущей даты возьмём дату рождения самого младшего ребёнка. Сосчитаем суммы возрастов родителей и детей на эту дату, занесём их в переменные *sumP* и *sumC* соответственно.

В цикле будем увеличивать текущую дату по одному дню. Во вложенном цикле пройдемся по массиву дат рождения родителей и увеличим значение *sumP* для каждого родителя, чей день рождения пришёлся на текущую дату. Аналогично, пройдемся по массиву дат рождения детей и увеличим значение *sumC* для каждого родителя, чей день рождения пришёлся на текущую дату. Отметим, что если текущая дата равна 1 марта невисокосного года, то нужно ещё учитывать детей и родителей, родившихся 29 февраля.

Как только *sumC* станет больше или равен *sumP*, ответ найден.

Оценим объём вычислений в этом решении. По условию задачи, все даты лежат в XX или XXI веке. Тогда в худшем случае интервал между датами рождения детей и родителей составит около 200 лет (мы не учитываем в задаче реальность такой ситуации). В худшем случае в семье всего три ребёнка. Тогда получается, что искомая дата наступит примерно через 400 лет, или примерно через 146 тысяч дней, через которые мы должны пройти, пока не дойдём до ответа.

Пример решения на Free Pascal:

```

uses SysUtils;

const
  NMAX = 10;
  dm: array[1..12] of longint = (31, 28, 31, 30, 31, 30, 31,
    31, 30, 31, 30, 31);

type
  Date = record
    d, m, y: longint;
  end;

function isLeap(y: longint): boolean;
begin
  isLeap := (y mod 4 = 0) and (y mod 100 <> 0)
    or (y mod 400 = 0);
end;

function mDays(m, y: longint): longint;
begin
  if m <> 2 then mDays := dm[m]
  else
    if isLeap(y) then mDays := 29 else mDays := 28;
  end;
end;

procedure incDate(var dt: Date);
begin
  with dt do begin
    inc(d);
    if d > mDays(m, y) then begin
      inc(m);
      d := 1;
      if m > 12 then begin
        inc(y);
        m := 1;
      end;
    end;
  end;
end;

function Age(d1, d2: Date): longint;
begin
  if (d1.m < d2.m) or (d1.m = d2.m) and (d1.d <= d2.d) then
    Age := d2.y - d1.y
  else
    Age := d2.y - d1.y - 1;
  end;
end;

```

```

function readDate: Date;
var
  dt: Date;
  s: String[10];
begin
  readln(s);
  dt.d := (ord(s[1]) - ord('0')) * 10 + ord(s[2]) - ord('0');
  dt.m := (ord(s[4]) - ord('0')) * 10 + ord(s[5]) - ord('0');
  dt.y := ((ord(s[7]) - ord('0')) * 10 + ord(s[8]) -
ord('0')) * 10 + ord(s[9]) - ord('0') * 10 + ord(s[10]) -
ord('0');
  readDate := dt;
end;

function less(a, b: Date): boolean;
begin
  less := (a.y < b.y)
    or (a.y = b.y) and (a.m < b.m)
    or (a.y = b.y) and (a.m = b.m) and (a.d < b.d);
end;

function isBirthday(dBirth, dCur: Date): boolean;
begin
  isBirthday := false;
  if (dBirth.m = dCur.m) and (dBirth.d = dCur.d) or
    (dBirth.m = 2) and (dBirth.d = 29) and (dCur.m = 3) and
    (dCur.d = 1) and not isLeap(dCur.y) then
    isBirthday := true;
end;

var
  n, sumP, sumC, i : longint;
  dp: array[1..2] of Date;
  dc: array[1..NMAX] of Date;
  cur: Date;

begin
  readln(n);
  if n < 3 then begin
    writeln(-1);
    exit;
  end;
  cur.d := 1; cur.m := 1; cur.y := 1900;
  for i := 1 to 2 do
    dp[i] := readDate;
  cur := dp[1];
  for i := 1 to n do begin
    dc[i] := readDate;
    if less(cur, dc[i]) then cur := dc[i];
  end;
  sumP := 0;

```

```

for i := 1 to 2 do
  inc(sumP, Age(dp[i], cur));
sumC := 0;
for i := 1 to n do
  inc(sumC, Age(dc[i], cur));
repeat
  incDate(cur);
  for i := 1 to 2 do
    if isBirthday(dp[i], cur) then
      inc(sumP);
  for i := 1 to n do
    if isBirthday(dc[i], cur) then
      inc(sumC);
until sumC >= sumP;
writeln(Format('%.2d.%.2d.%.2d', [cur.d, cur.m, cur.y]));
end.

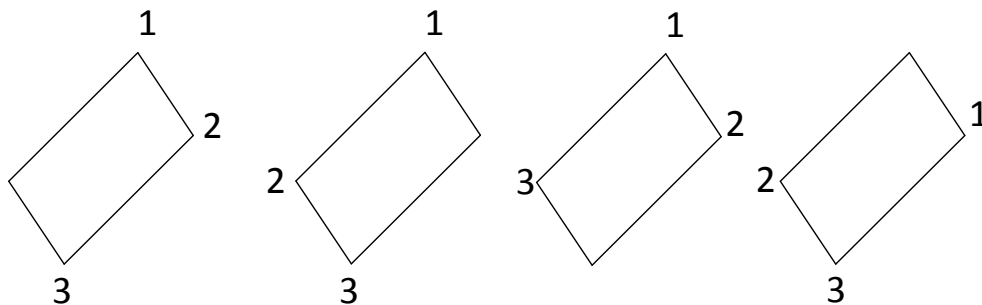
```

Задача 4. Параллелограммы (100 баллов)

Задача представляет собой изменённую и упрощённую версию задачи "Конкурс капитанов" для 9-11 классов.

Для удобства занесём в отдельный массив координаты всех крестиков (хотя можно этого и не делать, а работать прямо с матрицей входных данных). С помощью трёх вложенных циклов будем перебирать координаты трёх возможных вершин параллелограмма в порядке обхода. Зная координаты трёх вершин, вычислим координаты четвёртой. Если эти координаты не выходят за пределы поля и в клетке стоит крестик, то очередной кандидат в параллелограммы найден. Проверим, не является ли он вырожденным (например, сосчитав площадь и сравнив её с нулём). Если параллелограмм не вырожденный, то увеличим счётчик найденных параллелограммов на единицу.

В этом решении есть одна проблема (которой нет в версии задачи для 9-11 классов). Дело в том, что один и тот же параллелограмм может посчитаться несколько раз. Чтобы понять, сколько раз будет сосчитан один и тот же параллелограмм, рассмотрим рисунок.



На нём показано, что наш перебор может посетить три вершины одного и того же параллелограмма четырьмя разными случаями.

Однако, в нашем решении мы ищем вершины обязательно в порядке обхода, то есть последние два случая нам не подходят. Таким образом, каждый параллелограмм посчитается только дважды, то есть ответ нужно поделить на два.

Пример решения на Free Pascal:

```
const
  MAXSIZE = 25;

type
  Coord = record
    x, y: longint;
  end;

var
  a: array[1 .. MAXSIZE] of String;
  c: array[1 .. MAXSIZE * MAXSIZE] of Coord;
  ySize, xSize, i, j, k, n, x4, y4, ans: longint;

procedure FillCoord;
var
  y, x: longint;
begin
  n := 0;
  for y := 1 to ySize do
    for x := 1 to xSize do
      if a[y][x] = 'x' then begin
        inc(n);
        c[n].y := y;
        c[n].x := x;
      end;
    end;
  end;
end;

begin
  readln(ySize, xSize);
  for i := 1 to ySize do begin
    readln(a[i]);
  end;
  FillCoord;
  ans := 0;
  for i := 1 to n - 2 do
    for j := i + 1 to n - 1 do
      for k := j + 1 to n do begin
        x4 := c[i].x + c[k].x - c[j].x;
        y4 := c[i].y + c[k].y - c[j].y;
        if (x4>=1) and (x4<=xSize) and (y4>=1)
          and (y4<=ySize) then
```

```
        if (a[y4][x4] = 'x')  
            and ((c[i].x - c[j].x) * (c[k].y - c[j].y) -  
(c[k].x - c[j].x) * (c[i].y - c[j].y) <> 0) then  
                inc(ans);  
            end;  
        writeln(ans div 2);  
    end.
```

Для решения только первой подзадачи можно перебирать все четыре вершины четырьмя вложенными циклами и проверять, образуют ли они параллелограмм.