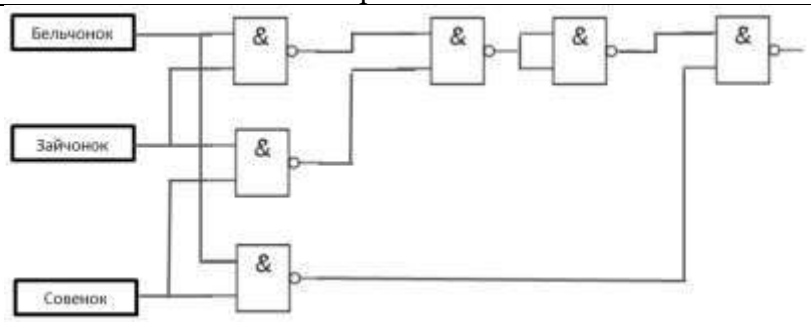
Информатика. 11 класс
Решения и ответы

1 вариант

<table>
<tr><th>№</th><th>Правильный ответ</th><th>Балл</th><th>Прим</th></tr>
<tr><td>1.</td><td> 
</td><td>15</td><td></td></tr>
<tr><td>2.</td><td>Один из вариантов верного алгоритма

Сначала Робот двигается вправо до начала кучи. Затем двигается вправо до первого гриба (если при этом нашли пустую ячейку вместо гриба – это и будет разделитель между грибами и шишками, то алгоритм завершается), убирает его, двигается вправо до первой пустой ячейки и кладет туда гриб. Затем двигается влево до первой шишки, убирает ее и двигается влево до первой пустой ячейки. Очевидно, что это будет та ячейка, где до этого был гриб, который убрал Робот. Записываем в данную ячейку шишку и заново начинаем поиск гриба движением вправо.

<table>
<tr><td>1.</td><td>? 3 2 2</td><td rowspan="2">Двигаемся вправо, пока не дойдем до начала кучи</td></tr>
<tr><td>2.</td><td>> 1</td></tr>
<tr><td>3.</td><td>? 17 5 4</td><td rowspan="2">Идем до первого Гриба. Если нашли пустую ячейку, то алгоритм завершен (переходим на конец программы)</td></tr>
<tr><td>4.</td><td>> 3</td></tr>
<tr><td>5.</td><td>Н 6</td><td rowspan="2">Заменяем Гриб на Ничего и сдвигаемся вправо</td></tr>
<tr><td>6.</td><td>> 7</td></tr>
<tr><td>7.</td><td>? 9 8 8</td><td rowspan="2">Идем вправо до первой пустой ячейки. Это будет левый край кучи</td></tr>
<tr><td>8.</td><td>> 7</td></tr>
<tr><td>9.</td><td>Г 10</td><td>Записываем в найденную ячейку Гриб</td></tr>
<tr><td>10.</td><td>? 17 11 12</td><td rowspan="2">Двигаемся влево до первой шишки. Если нашли пустую ячейку, то алгоритм завершен (переходим на конец программы)</td></tr>
<tr><td>11.</td><td>< 10</td></tr>
<tr><td>12.</td><td>Н 13</td><td rowspan="2">Заменяем Шишку на Ничего и сдвигаемся влево</td></tr>
<tr><td>13.</td><td>< 14</td></tr>
<tr><td>14.</td><td>? 16 15 15</td><td rowspan="2">Двигаемся влево до пустой ячейки (там где был до этого Гриб)</td></tr>
<tr><td>15.</td><td>< 14</td></tr>
<tr><td>16.</td><td>Ш 3</td><td>Записываем в найденную пустую ячейку Шишку и повторяем алгоритм поиска Гриба</td></tr>
<tr><td>17.</td><td></td><td>Конец программы</td></tr>
</table>
</td><td>30</td><td></td></tr>
</table>

<table>
<tr>
<td>3.</td>
<td>Для определения наибольшего возможного значения заряда батареи Робота следует хранить наибольшие значения, которые могут быть получены ходом слева и ходом справа. Большим по модулю отрицательным числом помечаются ячейки, в которые Робот не может попасть данным ходом.
Ниже показан пример расчета для поля размером 4x4
<table>
<tr><th></th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th><th>G</th><th>H</th><th>I</th><th>J</th><th>K</th><th>L</th><th>M</th><th>N</th></tr>
<tr><td>1</td><td colspan="4">Исходные данные</td><td></td><td colspan="4">Макс. значение (ход слева)</td><td></td><td colspan="4">Макс. значение (ход сверху)</td></tr>
<tr><td>2</td><td>15</td><td>0</td><td>10</td><td>10</td><td></td><td>15</td><td>5</td><td>5</td><td>5</td><td></td><td>15</td><td>-9999</td><td>-9999</td><td>-9999</td></tr>
<tr><td>3</td><td>5</td><td>4</td><td>150</td><td>5</td><td></td><td>-9999</td><td>-9999</td><td>-9999</td><td>90</td><td></td><td>10</td><td>-9999</td><td>100</td><td>-9999</td></tr>
<tr><td>4</td><td>10</td><td>10</td><td>10</td><td>10</td><td></td><td>-9999</td><td>5</td><td>5</td><td>95</td><td></td><td>10</td><td>-9999</td><td>100</td><td>85</td></tr>
<tr><td>5</td><td>30</td><td>10</td><td>10</td><td>10</td><td></td><td>-9999</td><td>25</td><td>25</td><td>95</td><td></td><td>30</td><td>-9999</td><td>100</td><td>90</td></tr>
</table>

</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Пример программы, выдающей верные результаты на всех тестах

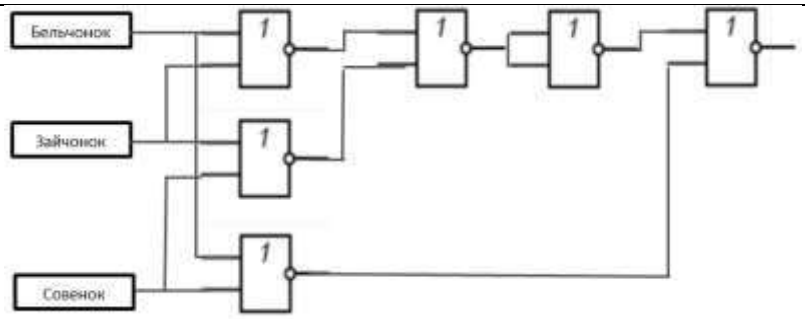
```
def game(x, m):
    used.add(x)
    for i in sm[x]:
        if i != m:
            if i not in used:
                game(i, x)
            if wins[i] == 'F':
                wins[x] = 'W'
    if len(sm[x]) == 1 and sm[x][0] == m:
        wins[x] = 'W'
    if wins[x] == 0:
        wins[x] = 'F'


used = set()
n, k = map(int, input().split())
wins = [0 for i in range(n + 1)]
sm = [[] for i in range(n + 1)]
for i in range(n - 1):
    a, b = map(int, input().split())
    sm[a].append(b)
    sm[b].append(a)
game(k, 0)
if wins[k] == 'W':
    sm[k].sort()
    for i in sm[k]:
        if wins[i] == 'F':
            print('B',i)
            break
else:
    print('M')
```

</td>
<td>30</td>
<td></td>
</tr>
</table>

5.	Так как известно, что многоугольники не пересекаются, то их можно отсортировать по какой-либо одной минимальной (максимальной) координате.
	Пример программы, выдающей верные результаты на всех тестах
	
	`mn = []`
	`g_all, r_all = 0, 0`
	`for _ in range(int(input())):`
	`  a = list(input().split())`
	`  color = a[0]`
	`  if color == 'G':`
	`  g_all += 1`
	`  else:`
	`  r_all += 1`
	`  y = []`
	`  for i in range(2, len(a), 2):`
	`  y.append(int(a[i]))`
	`  y.sort()`
	`  mn.append([y[-1], color])`
	`mn.sort()`
	`ans = 0`
	`g, r = 0, 0`
	`for i in mn:`
	`  if i[1] == 'R':`
	`  if g > 0 and g < g_all:`
	`  ans += 1`
	`  r += 1`
	`  elif i[1] == 'G':`
	`  g += 1`
	
	`print(ans)`
		20	

Информатика. 11 класс
Решения и ответы

2 вариант

<table>
<tr><th>№</th><th>Правильный ответ</th><th>Балл</th><th>Прим</th></tr>
<tr><td>1.</td><td> 
</td><td>15</td><td></td></tr>
<tr><td>2.</td><td>Идея алгоритма: меняем по одному шишки на грибы, а грибы на шишки. При этом, если число грибов и шишек не совпадает, то в одной из куч останутся и грибы и шишки. Теперь продолжая менять предметы в большей куче, дописываем недостающие во вторую. После завершения алгоритма кучи будут равны, но переставлены местами. Остается заменить все грибы на шишки и наоборот.
<table>
<tr><td>1.</td><td>? 2 3 3</td><td rowspan="4">Двигаемся вправо, пока не дойдем пустой ячейки, которая разделяет кучи шишек и грибов. Будем называть эту ячейку разделителем</td></tr>
<tr><td>2.</td><td>> 1</td></tr>
<tr><td>3.</td><td>? 5 4 4</td></tr>
<tr><td>4.</td><td>> 3</td></tr>
<tr><td>5.</td><td>? 16 6 7</td><td rowspan="2">Идем влево, пока не встретим шишку. Если все шишки закончились, то переходим к строке 16</td></tr>
<tr><td>6.</td><td>< 5</td></tr>
<tr><td>7.</td><td>Г ></td><td>Встретили шишку, меняем ее на гриб</td></tr>
<tr><td>8.</td><td>? 10 9 9</td><td rowspan="2">Идем вправо до разделителя</td></tr>
<tr><td>9.</td><td>> 8</td></tr>
<tr><td>10.</td><td>? 26 11 11</td><td rowspan="2">Идем вправо до первого гриба. Если все грибы закончились, то переходим к строке 26</td></tr>
<tr><td>11.</td><td>> 8</td></tr>
<tr><td>12.</td><td>Ш <</td><td>Встретили гриб, меняем его на шишку</td></tr>
<tr><td>13.</td><td>? 5 14 14</td><td rowspan="2">Идем влево до разделителя.</td></tr>
<tr><td>14.</td><td>< 13</td></tr>
<tr><td>15.</td><td>< 5</td><td>Дошли до разделителя, то повторяем алгоритм, начиная с 5 строки</td></tr>
<tr><td>16.</td><td>? 10 17 17</td><td rowspan="2">Попадаем сюда, если закончились шишки (см. строку 5). Идем вправо до разделителя</td></tr>
<tr><td>17.</td><td>> 16</td></tr>
<tr><td>18.</td><td>? 36 9 9</td><td rowspan="2">Идем вправо до первого гриба. Если гриб не найден, то кучи равны по количеству и переходим к замене грибов на шишки и шишек на грибы.</td></tr>
<tr><td>19.</td><td>> 18</td></tr>
<tr><td>20.</td><td>Ш <</td><td>Встретили гриб, меняем его на шишку</td></tr>
</table></td><td>30</td><td></td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>21.</td><td>? 5 22 22</td><td rowspan="2">Идем влево до разделителя.</td></tr>
<tr><td>22.</td><td>< 21</td></tr>
<tr><td>23.</td><td>? 5 24 24</td><td rowspan="2">Идем влево до первой пустой ячейки.</td></tr>
<tr><td>24.</td><td>< 23</td></tr>
<tr><td>25.</td><td>Г 18</td><td>Добавляем один гриб слева и повторяем алгоритм</td></tr>
<tr><td>26.</td><td>? 28 27 27</td><td rowspan="2">Попадаем сюда, если закончились грибы (см. строку 10). Идем вправо до разделителя</td></tr>
<tr><td>27.</td><td>> 16</td></tr>
<tr><td>28.</td><td>? 36 9 9</td><td rowspan="2">Идем вправо до первой шишки. Если шишка не найдена, то кучи равны по количеству и переходим к замене грибов на шишки и шишек на грибы.</td></tr>
<tr><td>29.</td><td>> 18</td></tr>
<tr><td>30.</td><td>Г <</td><td>Встретили шишку, меняем его на гриб</td></tr>
<tr><td>31.</td><td>? 33 32 32</td><td rowspan="2">Идем вправо до разделителя.</td></tr>
<tr><td>32.</td><td>< 21</td></tr>
<tr><td>33.</td><td>? 35 34 34</td><td rowspan="2">Идем вправо до первой пустой ячейки.</td></tr>
<tr><td>34.</td><td>< 23</td></tr>
<tr><td>35.</td><td>Г 26</td><td>Добавляем одну шишку справа и повторяем алгоритм</td></tr>
<tr><td>36.</td><td>? 41 37 39</td><td rowspan="5">Делаем замену шишке на грибы и грибы на шишки</td></tr>
<tr><td>37.</td><td>Ш 38</td></tr>
<tr><td>38.</td><td>> 36</td></tr>
<tr><td>39.</td><td>Г 40</td></tr>
<tr><td>40.</td><td>> 36</td></tr>
<tr><td>41.</td><td></td><td>Завершение программы</td></tr>
</table>
</td><td></td><td></td></tr>
<tr><td>3.</td><td>Для определения наибольшего возможного значения заряда батареи Робота следует хранить наибольшие значения, которые могут быть получены ходом справа и ходом снизу. Большим по модулю отрицательным числом помечаются ячейки, в которые Робот не может попасть данным ходом. В качестве ответа необходимо указать наибольшее из значений, полученных в финишной ячейке.
Ниже показан пример расчета для поля размером 4x4
<table>
<tr><td></td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td></tr>
<tr><td>1</td><td colspan="4">Исходные данные</td><td></td><td colspan="4">Макс. значение (ход справа)</td><td></td><td colspan="4">Макс. значение (ход снизу)</td></tr>
<tr><td>2</td><td>15</td><td>70</td><td>1</td><td>3</td><td></td><td>86</td><td>81</td><td>-9999</td><td>-9999</td><td></td><td>30</td><td>79</td><td>26</td><td>-9999</td></tr>
<tr><td>3</td><td>5</td><td>4</td><td>15</td><td>5</td><td></td><td>19</td><td>24</td><td>1</td><td>-9999</td><td></td><td>25</td><td>9</td><td>35</td><td>1</td></tr>
<tr><td>4</td><td>10</td><td>5</td><td>30</td><td>1</td><td></td><td>20</td><td>20</td><td>21</td><td>-9999</td><td></td><td>30</td><td>5</td><td>30</td><td>6</td></tr>
<tr><td>5</td><td>30</td><td>10</td><td>10</td><td>15</td><td></td><td>35</td><td>15</td><td>15</td><td>15</td><td></td><td>-9999</td><td>-9999</td><td>-9999</td><td>15</td></tr>
</table>
 </td><td>5</td><td></td></tr>
<tr><td>4.</td><td>Пример программы, выдающей верные результаты на всех тестах

<pre>def game(x, m):</pre></td><td>30</td><td></td></tr>
</table>

```
    used.add(x)
    for i in sm[x]:
        if i != m:
            if i not in used:
                game(i, x)
            if wins[i] == 'F':
                wins[x] = 'W'
    if len(sm[x]) == 1 and sm[x][0] == m:
        wins[x] = 'W'
    if wins[x] == 0:
        wins[x] = 'F'


used = set()
n, k = map(int, input().split())
wins = [0 for i in range(n + 1)]
sm = [[] for i in range(n + 1)]
for i in range(n - 1):
    a, b = map(int, input().split())
    sm[a].append(b)
    sm[b].append(a)
game(k, 0)
if wins[k] == 'W':
    sm[k].sort()
    for i in sm[k]:
        if wins[i] == 'F':
            print('B',i)
            break
else:
    print('Z')
```

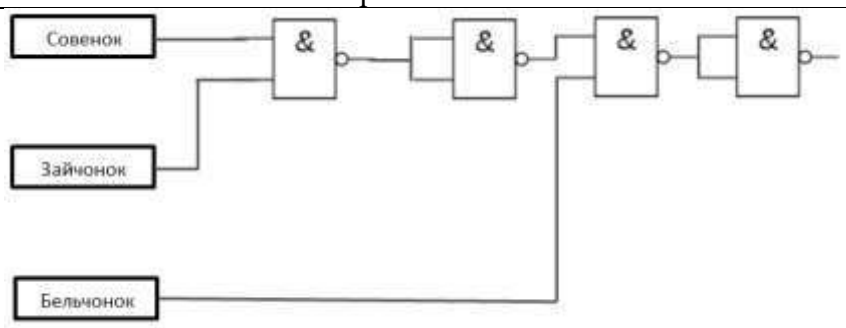
5. Так как известно, что многоугольники не пересекаются, то их можно отсортировать по какой-либо одной минимальной (максимальной) координате.
Пример программы, выдающей верные результаты на всех тестах

20

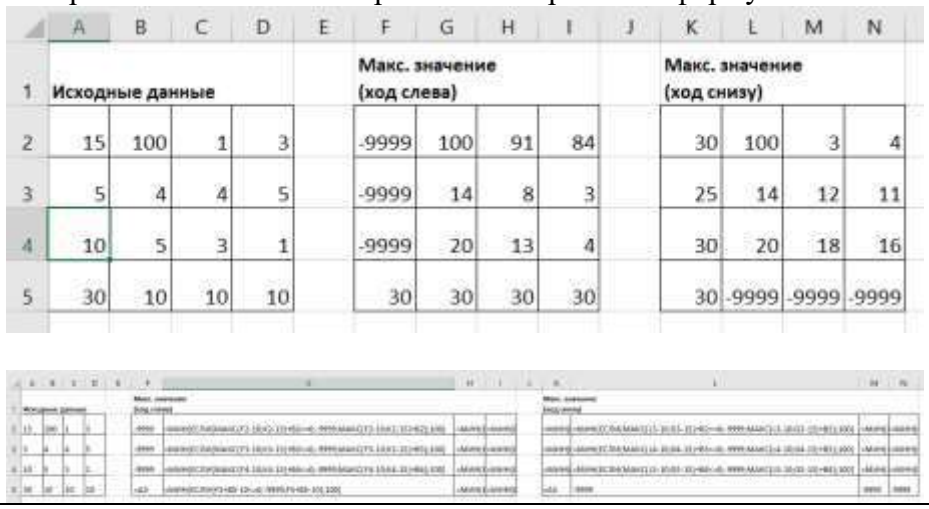
```
mn = []
for _ in range(int(input())):
    a = list(input().split())
    color = a[0]
    y = []
    for i in range(2, len(a), 2):
        y.append(int(a[i]))
    y.sort()
    mn.append([y[-1], color, len(y)])
mn.sort()

print(mn[0][2], mn[0][1], ' ', mn[-1][2], mn[-
1][1], sep="")
```

Информатика. 11 класс
Решения и ответы

3 вариант

<table>
<tr><th>№</th><th>Правильный ответ</th><th>Балл</th><th>Прим</th></tr>
<tr><td>1.</td><td> 
</td><td>15</td><td></td></tr>
<tr><td>2.</td><td>Один из вариантов верного алгоритма для Робота

Сначала Робот двигается влево до начала кучи. Затем двигается влево до первого гриба (если при этом нашли пустую ячейку вместо гриба – это и будет разделитель между грибами и шишками, то алгоритм завершается), убирает его, двигается влево до первой пустой ячейки и кладет туда гриб. Затем двигается вправо до первой шишки, убирает ее и двигается вправо до первой пустой ячейки. Очевидно, что это будет та ячейка, где до этого был гриб, который убрал Робот. Записываем в данную ячейку шишку и заново начинаем поиск гриба движением влево.
<table>
<tr><td>1.</td><td>? 3 2 2</td><td rowspan="2">Двигаемся влево, пока не дойдем до начала кучи</td></tr>
<tr><td>2.</td><td>< 1</td></tr>
<tr><td>3.</td><td>? 17 5 4</td><td rowspan="2">Идем до первого Гриба. Если нашли пустую ячейку, то алгоритм завершен (переходим на конец программы)</td></tr>
<tr><td>4.</td><td>< 3</td></tr>
<tr><td>5.</td><td>Н 6</td><td rowspan="2">Заменяем Гриб на Ничего и сдвигаемся влево</td></tr>
<tr><td>6.</td><td>< 7</td></tr>
<tr><td>7.</td><td>? 9 8 8</td><td rowspan="2">Идем влево до первой пустой ячейки. Это будет левый край кучи</td></tr>
<tr><td>8.</td><td>< 7</td></tr>
<tr><td>9.</td><td>Г 10</td><td>Записываем в найденную ячейку Гриб</td></tr>
<tr><td>10.</td><td>? 17 11 12</td><td rowspan="2">Двигаемся вправо до первой шишки слева. Если нашли пустую ячейку, то алгоритм завершен (переходим на конец программы)</td></tr>
<tr><td>11.</td><td>> 10</td></tr>
<tr><td>12.</td><td>Н 13</td><td rowspan="2">Заменяем Шишку на Ничего и сдвигаемся вправо</td></tr>
<tr><td>13.</td><td>> 14</td></tr>
<tr><td>14.</td><td>? 16 15 15</td><td rowspan="2">Двигаемся вправо до пустой ячейки (там где был до этого Гриб)</td></tr>
<tr><td>15.</td><td>> 14</td></tr>
<tr><td>16.</td><td>Ш 3</td><td>Записываем в найденную пустую ячейку Шишку и повторяем алгоритм поиска Гриба</td></tr>
<tr><td>17.</td><td></td><td>Конец программы</td></tr>
</table></td><td>30</td><td></td></tr>
</table>

<table>
<tr>
<td>3.</td>
<td>Для определения наибольшего возможного значения заряда батареи Робота следует хранить наибольшие значения, которые могут быть получены ходом слева и ходом снизу. Большим по модулю отрицательным числом помечаются ячейки, в которые Робот не может попасть данным ходом.
Ниже показан пример расчета для поля размером 4x4 в режиме отображения значений и в режиме отображения формул
 
<table>
<tr><td></td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td><td>J</td><td>K</td><td>L</td><td>M</td><td>N</td></tr>
<tr><td>1</td><td colspan="4">Исходные данные</td><td></td><td colspan="4">Макс. значение (ход слева)</td><td></td><td colspan="4">Макс. значение (ход снизу)</td></tr>
<tr><td>2</td><td>15</td><td>100</td><td>1</td><td>3</td><td></td><td>-9999</td><td>100</td><td>91</td><td>84</td><td></td><td>30</td><td>100</td><td>3</td><td>4</td></tr>
<tr><td>3</td><td>5</td><td>4</td><td>4</td><td>5</td><td></td><td>-9999</td><td>14</td><td>8</td><td>3</td><td></td><td>25</td><td>14</td><td>12</td><td>11</td></tr>
<tr><td>4</td><td>10</td><td>5</td><td>3</td><td>1</td><td></td><td>-9999</td><td>20</td><td>13</td><td>4</td><td></td><td>30</td><td>20</td><td>18</td><td>16</td></tr>
<tr><td>5</td><td>30</td><td>10</td><td>10</td><td>10</td><td></td><td>30</td><td>30</td><td>30</td><td>30</td><td></td><td>30</td><td>-9999</td><td>-9999</td><td>-9999</td></tr>
</table>
</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Пример программы, выдающей верные результаты на всех тестах


```
def game(x, m):
    used.add(x)
    for i in sm[x]:
        if i != m:
            if i not in used:
                game(i, x)
            if wins[i] == 'F':
                wins[x] = 'W'
    if len(sm[x]) == 1 and sm[x][0] == m:
        wins[x] = 'W'
    if wins[x] == 0:
        wins[x] = 'F'


used = set()
n, k = map(int, input().split())
wins = [0 for i in range(n + 1)]
sm = [[] for i in range(n + 1)]
for i in range(n - 1):
    a, b = map(int, input().split())
    sm[a].append(b)
    sm[b].append(a)
game(k, 0)
if wins[k] == 'W':
    sm[k].sort()
    for i in sm[k]:
        if wins[i] == 'F':
            print('B',i)
            break
```

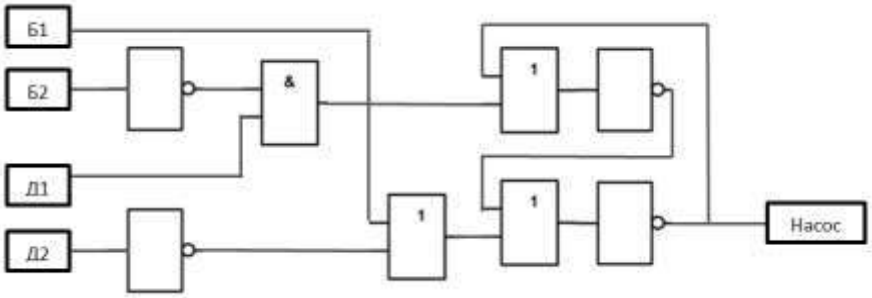
</td>
<td>30</td>
<td></td>
</tr>
</table>

	else: print('M')		

5.	Так как известно, что многоугольники не пересекаются, то их можно отсортировать по какой-либо одной минимальной (максимальной координате).
	Пример программы, выдающей верные результаты на всех тестах
	
	mn = []
	g_all, r_all = 0, 0
	for _ in range(int(input())):
	    a = list(input().split())
	    color = a[0]
	    if color == 'G':
	        g_all += 1
	    else:
	        r_all += 1
	    y = []
	    for i in range(2, len(a), 2):
	        y.append(int(a[i]))
	    y.sort()
	    mn.append([y[-1], color])
	mn.sort()
	ans = 0
	g, r = 0, 0
	for i in mn:
	    if i[1] == 'G':
	        if r > 0 and r < r_all:
	            ans += 1
	        g += 1
	    elif i[1] == 'R':
	        r += 1
	
	print(ans)
		20	

Информатика. 11 класс
Решения и ответы

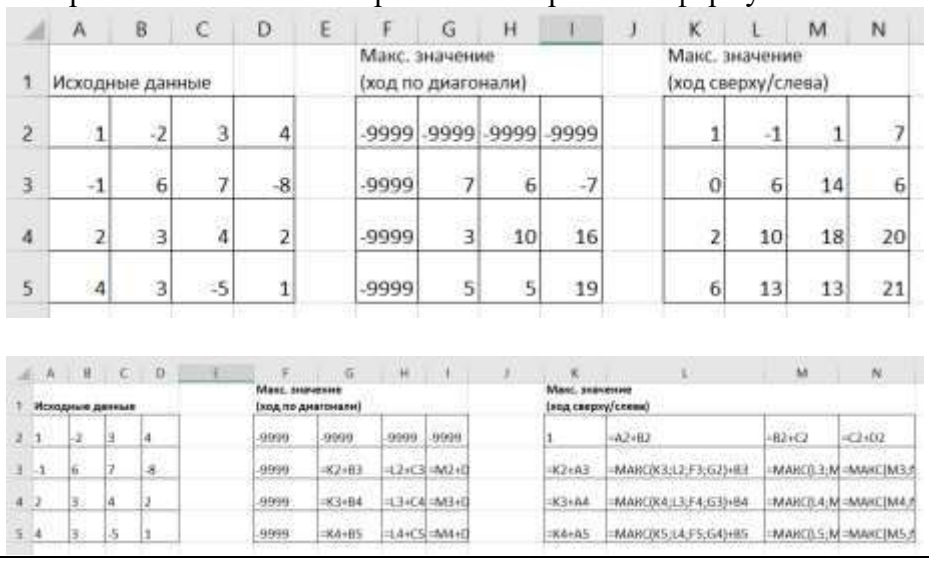
4 вариант

<table>
<tr><th>№</th><th>Правильный ответ</th><th>Балл</th><th>Прим</th></tr>
<tr><td>1.</td><td> 

Очевидно, что схема должна обладать эффектом памяти, так как при одних и тех же значениях датчиков насос может быть как включен, так и выключен в зависимости от предыдущего состояния системы. Поэтому в схему обязательно должен быть включен триггер.</td><td>15</td><td></td></tr>
<tr><td>2.</td><td>Один из вариантов верного алгоритма для Робота

Робот доходит до начала кучи, сдвигается на 1 ячейку вправо, берет вторую слева шишку и записывает ее в правый конец кучи. Затем он сдвигается влево на 1 ячейку, берет шишку (это будет вторая справа шишка), двигается на 1 ячейку влево и двигается до пустой ячейки (где до этого была шишка). Записывает шишку на место этой пустой ячейки, сдвигается на 1 позицию вправо, берет шишку (это будет третья шишка слева), двигается вправо до пустой ячейки, записывает туда шишку и т.д. Продолжая данные действия, Робот рано или поздно после того, как положит шишку и сдвинется на одну ячейку попадет в пустую ячейку. Это будет означать, что мы дошли до середины и алгоритм завершен

<table>
<tr><td>1.</td><td>? 2 3</td><td rowspan="2">Двигаемся вправо до начала кучи</td></tr>
<tr><td>2.</td><td>> 1</td></tr>
<tr><td>3.</td><td>> 4</td><td>Сдвигаемся на 1 ячейку вправо</td></tr>
<tr><td>4.</td><td>? 15 6</td><td>Если там (справа) пусто, алгоритм завершен</td></tr>
<tr><td>5.</td><td>Н 6</td><td rowspan="2">Убираем шишку и сдвигаемся вправо</td></tr>
<tr><td>6.</td><td>> 7</td></tr>
<tr><td>7.</td><td>? 8 6</td><td>Двигаемся вправо до пустой ячейки</td></tr>
<tr><td>8.</td><td>Ш 9</td><td rowspan="2">Записываем в найденную пустую ячейку шишку и сдвигаемся влево</td></tr>
<tr><td>9.</td><td>< 10</td></tr>
<tr><td>10.</td><td>? 15 11</td><td>Если там (слева) пусто, алгоритм завершен</td></tr>
<tr><td>11.</td><td>Н 12</td><td rowspan="2">Убираем шишку и сдвигаемся влево</td></tr>
<tr><td>12.</td><td>< 13</td></tr>
<tr><td>13.</td><td>? 14 12</td><td>Двигаемся влево до пустой ячейки</td></tr>
<tr><td>14.</td><td>Ш 3</td><td>Записываем туда шишку и повторяем шаги алгоритма, начиная с 3-го</td></tr>
</table></td><td>30</td><td></td></tr>
</table>

	15. | | Завершение программы		
3.	Для определения наибольшего возможного значения заряда батареи Робота следует хранить наибольшие значения, которые могут быть получены ходом по диагонали и ходом сверху/слева. Большим по модулю отрицательным числом помечаются ячейки, в которые Робот не может попасть данным ходом. Ниже показан пример расчета для поля размером 4x4 в режиме отображения значений и в режиме отображения формул 	5	

4.	Пример программы, реализующей верный алгоритм
	
	`f = open("dict.txt", "r")`
	`s = [x.strip() for x in f.readlines()]`
	
	`def can(s1, s2): #проверка, можно ли перейти от слова s1 к слову s2`
	`    if abs(len(s1) - len(s2)) == 1:`
	`        if s1 in s2 or s2 in s1:`
	`            return True`
	`        else:`
	`            return False`
	`    elif abs(len(s1) - len(s2)) > 1:`
	`        return False`
	`    elif len(s1) == len(s2) and s1 != s2:`
	`        f = False`
	`        for i in range(len(s1)):`
	`            t1 = s1[0:i] + s1[i + 1:]`
	`            t2 = s2[0:i] + s2[i + 1:]`
	`            if t1 == t2:`
	`                f = True`
	`        return f`
	`    elif s1 == s2:`
	`        return False`
	
	`wins = set()`
		30	

	`def game(gamer, history):`
	`  nxt = []`
	`  for w in s:`
	`  if w not in history and can(history[-1], w):`
	`  nxt.append(w)`
	`  if not nxt:`
	`  if gamer == 1:`
	`  wins.add(history[1])`
	`  return "F"`
	
	`  for w in nxt:`
	`  if game((gamer + 1)%2, history+[w]) == "F":`
	`  return "W"`
	`  return "F"`
	
	`start_word = input()`
	`res = game(0,[start_word])`
	`if res == "W":`
	`  print("Z", sorted(list(wins))[0])`
	`else:`
	`  print("B")`		

5.	Основной трудностью в данной задаче является написание функции проверки пересечения двух отрезков. В примере ниже проверка пересечения двух отрезков производится при помощи ориентированной площади треугольника. Возможны и другие варианты решения, например, при помощи построения уравнений прямой.
	
	`def area (ax, ay, bx, by, cx, cy):`
	`  return (bx - ax) * (cy - ay) - (by - ay) * (cx - ax)`
	
	`def intersect_1 (a, b, c, d):`
	`  if (a > b):  a,b = b,a`
	`  if (c > d):  c,d = d,c`
	`  return max(a,c) <= min(b,d)`
	
	`def intersect (ax, ay, bx, by, cx, cy, dx, dy):`
	`  return intersect_1 (ax, bx, cx, dx) and intersect_1 (ay, by, cy, dy) and\`
	`  area(ax, ay,bx,by,cx,cy) * area(ax,ay,bx,by,dx,dy) <= 0 and\`
	`  area(cx,cy,dx,dy,ax,ay) * area(cx,cy,dx,dy,bx,by) <= 0`
	
	`data = []`
	`n = int(input())`
	`for _ in range(n):`
	`  row = input().split()`
		20	

```
        data.append((row[0], int(row[1]),
int(row[2]), int(row[3]), int(row[4])))

k_max = 0

for i in range(len(data)):
        k = 0
        for j in range(len(data)):
                if data[i][0] != data[j][0] and
intersect(data[i][1],data[i][2],data[i][3],data
[i][4],data[j][1],data[j][2],data[j][3],data[j]
[4]):
                        k += 1
                        if k > k_max:
                                k_max = k
                                line_max =
data[i]
print(*line_max)
```

Информатика. 11 класс
Критерии оценивания

Задача 1.

Правильно составленная логическая схема, содержащая минимально возможное количество элементов, верно работающая при любых наборах входных данных – 15 баллов.

Правильно составленная логическая схема, содержащая большее количество элементов, чем минимально возможное, верно работающая при любых наборах входных данных – 10-13 баллов.

Приведено верное рассуждение, приводящие к верной логической формуле, содержащей только нужные элементы, в минимальном количестве, но при этом логическая схема не построена или построена неверно – 10 баллов.

Построенная схема верно работает только при некоторых наборах входных данных, при этом она содержит верные идеи – 3-8 баллов

Решение задачи содержит только отдельные элементы верного решения 1-2 балла

Задача 2.

Верно написанная программа для Робота с подробными пояснениями – 30 баллов

Изложен в целом верный алгоритм, программа содержит незначительные ошибки или недостаточно подробное описание – 25 баллов

Изложен в целом верный алгоритм, программа для Робота в целом верная, но при этом в ней не описано завершение алгоритма (после получения нужного результата, Робот продолжит действия) или программа содержит слишком много строк (реализован не оптимальный алгоритм)– 20 баллов.

Изложена верная идея алгоритма, при этом программа содержит ошибки, из-за которых работает не при всех входных данных 15-20 баллов.

Изложена верная идея алгоритма, при этом программа не написана или содержит существенные ошибки – 5 -14 баллов.

Решение задачи содержит только отдельные элементы верного решения 1-4 балла.

При отсутствии описания алгоритма на естественном языке максимальная оценка за задачу - 10 баллов.

Задача 3.

Произведен расчет ответа на основании формул. Осуществлено хранение всех необходимых предыдущих значений. Получен верный ответ – 5 баллов

Произведен расчет ответа на основании формул. Осуществлено хранение всех необходимых предыдущих значений, но получен неверный ответ – 4 балла

Верный ответ получен в результате «ручных» вычислений – 3 балла

Ответ неверный, решение имеет элементы верного решения – 1-2 балла

Задача 4.

Верное решение, выдающее верный ответ на всех предложенных тестах – 30 баллов

В целом верное решение, имеющие ошибки, приводящие в тому, что программа выдает верный ответ только на 2-х из 3-х тестов – 20 баллов

Решение, имеющие элементы верного алгоритма, выдающее верный ответ только на одном из тестов – 10 баллов

Задача 5.

Верное решение, выдающее верный ответ на всех предложенных тестах – 20 баллов

Верное решение, выдающее верный ответ на всех предложенных тестах, но при этом использующая более сложный алгоритм, чем возможно (например, вычисляются площади фигур) – 15-18 баллов

В целом верное решение, имеющие ошибки, приводящие в тому, что программа выдает верный ответ только на 2-х из 3-х тестов – 15 баллов

Решение, имеющие элементы верного алгоритма, выдающее верный ответ только на одном из тестов – 5 баллов