ФЕДЕРАЛЬНЫЙ ИНСТИТУТ ПЕДАГОГИЧЕСКИХ ИЗМЕРЕНИЙ

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ОЦЕНИВАНИЮ ЗАДАНИЙ С РАЗВЕРНУТЫМ ОТВЕТОМ

ИНФОРМАТИКА

Москва 2009

Научный руководитель: **Ковалева Г.С.,** заместитель директора ФИПИ

Авторы: **Андреева Е. В., Лещинер В. Р.,**
Самылкина Н. Н., Якушкин П. А. , Крылов С.С., Шедов С.В.

Авторы будут благодарны за замечания и предложения по совершенствованию пособия.

ОГЛАВЛЕНИЕ

Введение

Разработка системы единого государственного экзамена включает в себя создание большого количества взаимосвязанных подсистем. Одной из них является формирование комплекса стандартизированной подготовки экспертов-предметников, включающей эффективное обучение проверке заданий с развернутыми ответами контрольных измерительных материалов (в частности, по информатике) с точным соблюдением централизованно разработанных критериев оценивания выполнения учащимися заданий с развернутыми ответами. Решение этой задачи – одно из условий обеспечения объективности и надежности результатов, полученных в ходе единого государственного экзамена.

Предлагаемый Учебно-методический комплект (УМК) для подготовки экспертов привлекаемых для проверки заданий с развернутыми ответами по информатике в рамках ЕГЭ разработан на основе открытых вариантов КИМ ЕГЭ и анализа опыта подготовки экспертов последних двух лет. Предлагаемые материалы учитывают спецификацию экзамена 2009 года и состоят из двух частей:

Часть 1. Методика оценивания

Часть 2. Материалы для самостоятельной работы.

В первую часть вошли материалы, содержащие характеристики заданий группы С, подходы к оценке их решений; образцы решения задач и критерии оценки этих решений; примеры оценивания решений задач группы С, а также справочные материалы.

В тренингах для самостоятельной работы помещены решения учащихся для пошагового оценивания и для оценивания работ в целом.

Краткие описания заданий каждого типа и рекомендации по их оцениванию

Фрагменты спецификации экзаменационной работы по информатике 2009 года относящиеся к задачам раздела «С».

Общая информация

…

В 2009 г. проведение экзамена по информатике осуществляется в бескомпьютерном варианте. Для выполнения любого из заданий экзаменационной работы не требуется никакого дополнительного оборудования или программного обеспечения. Использование калькуляторов на экзамене по информатике запрещается.

Содержание заданий разработано по основным темам курса информатики и информационных технологий, объединенных в следующие тематические блоки: "Информация и её кодирование", "Алгоритмизация и программирование", "Основы логики", "Моделирование и компьютерный эксперимент", "Основные устройства информационных и коммуникационных технологий", "Программные средства информационных и коммуникационных технологий", "Технология обработки графической и звуковой информации", "Технология обработки информации в электронных таблицах", "Технология хранения, поиска и сортировки информации в базах данных", "Телекоммуникационные технологии".

Задания Части 3 направлены на проверку сформированности важнейших умений записи и анализа алгоритмов, предусмотренных требованиями к обязательному уровню подготовки по информатике учащихся средних общеобразовательных учреждений. Эти умения проверяются на повышенном и высоком уровне сложности. Также на высоком уровне сложности проверяются умения по теме "Технология программирования".

Распределение заданий экзаменационной работы по уровню сложности

…

В части 3 (С) всего четыре задания, относящиеся к повышенному и высокому уровню сложности.

Если для заданий базового уровня предполагаемый процент выполнения 60-80%, то к аданиям повышенного и высокого уровня сложности требования иные и они достаточно сильно различаются. Для задания повышенного уровня С1 предполагаемый процент выполнения от 40 до 60%, а для остальных задач части С (С2, С3, С4) предполагаемый процент выполнения от 10% до 30%.

…

Система оценивания отдельных заданий и экзаменационной работы в целом

…

Выполнение заданий Части 3 (С) оценивается от двух до четырех баллов.

Ответы на задания Части 3 (С) проверяются и оцениваются экспертами (устанавливается соответствие ответов определенному перечню критериев).

Максимальное количество баллов, которое можно получить за выполнение заданий Части 3 (С), – 12.

…

Фрагмент обобщенного плана экзаменационной работы по информатике для выпускников средней (полной) общеобразовательной школы 2008 г.

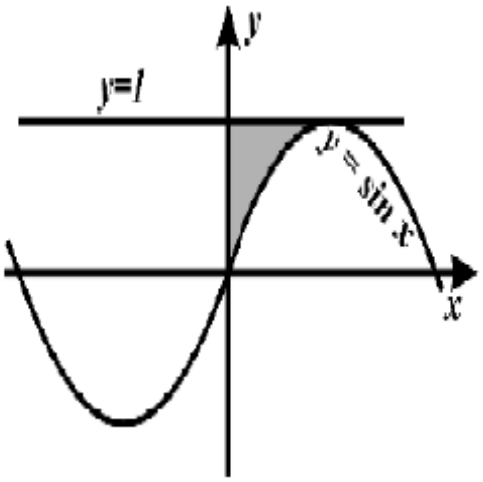
№ п/п	Обозначение задания	Проверяемые элементы содержания и виды деятельности	Коды проверяемых элементов	Уровень сложности задания	Тип задания	Максимальный балл за задание	Примерное время выполнения
	…	…	…	…	…	...	...
		Часть 3					
29	С1	**Умение прочесть фрагмент программы на языке программирования и исправить допущенные ошибки**	2.8.1./ 2.8.2.	П	РО	3	30
31	С2	**Умения написать короткую (10-15 строк) простую**	1.2.4/ 2.8.3	В	РО	2	30

		программу обработки массива на языке программирования или записать алгоритм на естественном языке					
32	C3	**Умение построить дерево игры по заданному алгоритму и обосновать выигрышную стратегию**	1.2.1	B	PO	3	30
37	C4	**Умения создавать собственные программы (30-50 строк) для решения задач средней сложности**	2.8.3.	B	PO	4	60
	…	…	…	…	…	...	...

Рекомендации по оцениванию заданий типа С1

Задача С1. Вариант 1.

Текст задачи:



Требовалось написать программу, которая вводит с клавиатуры координаты точки на плоскости (*x,y* – действительные числа) и определяет принадлежность точки заштрихованной области, включая ее границы. Программист торопился и написал программу неправильно.

ПРОГРАММА НА ПАСКАЛЕ	ПРОГРАММА НА БЕЙСИКЕ	ПРОГРАММА НА СИ

var x,y: real;
begin
readln(x,y);
if y<=1 then
if x>=0 then
if y>=sin(x) then
write('принадлежит')
else
write('не принадлежит')
end.
	INPUT x, y
	IF y<=1 THEN
	IF x>=0 THEN
	IF y>=SIN(x) THEN
	PRINT "принадлежит"
	ELSE
	PRINT "не принадлежит"
	ENDIF
	ENDIF
	ENDIF
	END
		void main(void)
		{ float x,y;
		scanf("%f%f",&x,&y);
		if (y<=1)
		if (x>=0)
		if (y>=sin(x))
		printf("принадлежит");
		else
		printf("не принадлежит");
		}

Последовательно выполните следующее:

1) Приведите пример таких чисел *x*, *y*, при которых программа неверно решает поставленную задачу.

2) Укажите, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы.)

Ответ (критерии оценивания):

Содержание верного ответа и указания по оцениванию **(допускаются иные формулировки ответа, не искажающие его смысла)**	

Элементы ответа:
1) Пример: x=3, y=0,5 (Любая пара (x,y), для которой выполняется: y>1 или x<0 или (y>=sin x и x>π/2 и y<=1))
2) Возможная доработка (Паскаль):
if (y<=1) and (x>=0) and (y>=sin(x)) and (x<=3,14/2) then
write('принадлежит')
else
write('не принадлежит')
(могут быть и другие способы доработки).	

Указания по оцениванию	**Баллы**
Обратите внимание! В задаче требовалось выполнить **три** действия: • указать пример входных данных, при которых программа работает неверно, и исправить две ошибки: • Неправильное использование условного оператора, в результате чего при невыполнении первого или второго условия программа не выдавала ничего (отсутствуют случаи ELSE). • Приведенным трем ограничениям удовлетворяют также те точки плоскости, у которых (y>=sin x и x>=π/2 и y<=1).	
Правильно выполнены оба пункта задания. Исправлены обе ошибки. Допускается замена числа π на 3,14 или другую константу. В работе (во фрагментах программ) допускается не более одной синтаксической ошибки	3

Правильно выполнены 2 пункта задания из трех (исправлены обе ошибки, но не указан/неправильно указан пример требуемых входных данных, либо правильно указан пример входных данных, программа правильно работает при большем числе случаев, чем исходная, но не при всех).
Например, выдает "принадлежит" для точек, у которых
(y>=sin x и x>π/2 и y<=1).
Допускается, например, такое решение:
if y<=1 then
if x>=0 then
if y>=sin(x) then
write('принадлежит')
else
write('не принадлежит')
else
write('не принадлежит')
else
write('не принадлежит')
При этом в сданной работе допускается не более двух синтаксических ошибок (пропущен или неверно указан знак пунктуации, неверно написано зарезервированное слово языка программирования).
	2

Правильно выполнен только один пункт задания. То есть, только приведен пример входных данных, либо он не приведен, но имеется программа, корректно работающая при большем количестве входных данных, чем исходная. При этом, если приведена программа, то в ней допускается не более трех синтаксических ошибок (пропущен или неверно указан знак пунктуации, неверно написано зарезервированное слово языка программирования)	1
Все пункты задания выполнены неверно (пример входных данных не указан или указан неверно, программа не приведена, либо приведенная программа корректно работает в не большем количестве случаев, чем исходная)	0
Максимальный балл	3

Задача С1. Вариант 2.

Текст задачи:

Требовалось написать программу, в которой нужно было проверить, лежит ли число **x** на числовой оси между числами **a** и **b** ("между" понимается в строгом смысле, т.е. случай **x=a** или **x=b** недопустим). Числа **x, a, b** являются натуральными, и известно, что **a** отлично от **b** (но неизвестно: **a>b** или **b>a**). Входная информация вводится с клавиатуры, а на выходе должно быть сообщение вида *"x между a,b*" (если это действительно так), в противном случае никакой выходной информации не выдается. Программист торопился и написал программу некорректно.

1) Привести пример таких чисел **a, b, x**, при которых программа работает неправильно.

2) Указать, как нужно доработать программу, чтобы не было случаев ее неправильной работы. (Это можно сделать несколькими способами, поэтому можно указать любой способ доработки исходной программы).

3) Наложено дополнительное условие: доработанная программа не должна использовать логических операций AND или OR. Указать, как можно доработать программу, соблюдая это условие.

ПРОГРАММА НА ПАСКАЛЕ	ПРОГРАММА НА БЕЙСИКЕ

VAR a,b,x: integer;
p: integer;
BEGIN
readln(a,b,x);
if (a>x) AND (x>b) then
writeln('x между a,b');
END.
	CLS
	INPUT a, b, x
	IF (a>x) AND (x>b) THEN
	PRINT “x между a, b”
	END

Ответ (критерии оценивания):

Содержание верного ответа и указания по оцениванию (допускаются иные формулировки ответа, не искажающие его смысла)	**Баллы**
Элементы ответа: 1) Пример: a=1 x=2 b=3 2) Возможная доработка: if a<b then begin p:=a; a:=b; b:=p end; if (a>x) AND (x>b) then writeln(' x между a,b'); (могут быть и другие правильные способы доработки) 3) Возможная доработка без использования связок AND, OR: p:=(x-a)*(x-b); if p<0 then writeln(' x между a,b'); (могут быть и другие способы доработки без использования связок AND, OR) При оценке других вариантов доработки программы нужно проверять, что поставленная цель достигается	
Указания по оцениванию	
Правильно выполнены п.1) + п.3) задания (т.к. выполнение п.3 "покрывает" и пункт 2), или правильно выполнены все 3 пункта задания, при этом в работе (во фрагментах программ) допускается не более одной синтаксической ошибки	**3**
Правильно выполнены 2 пункта задания:1) + 2) или 2) + 3), (причем способы доработки в п.2 и п.3 различны) - при этом в сданной работе допускается не более двух синтаксических ошибок .	**2**
Правильно выполнен только один пункт задания, при этом, если это был п.2) или п.3), то в нем допускается не более двух синтаксических ошибок	**1**
Все пункты задания выполнены неверно	**0**
Максимальный балл	***3***

Рекомендации по оцениванию заданий типа С2

Задача С2. Вариант 1.

Текст задачи:

Опишите на русском языке или одном из языков программирования алгоритм подсчета числа элементов, равных максимальному, в числовом массиве из 30 элементов.

Ответ (критерии оценивания):

Задание С2 относится к высокому уровню сложности. За решение этого задания можно получить два балла. По содержанию проверяет умение формально описывать известный (изученный в школе) алгоритм на естественном языке или на языке программирования.

<table>
<tr><th colspan="3">Содержание правильного ответа</th></tr>
<tr><td colspan="3">Введем целочисленную переменную Max, в которую будем заносить максимальный из просмотренных элементов массива, и целочисленную переменную Num, в которой будем подсчитывать число повторений максимального элемента в просмотренной части массива. Занесем в Max значение первого элемента массива, в Num занесем 1.
В цикле до конца массива: сравниваем очередной элемент массива с текущим значением переменной Max, если он больше, то заносим его значение в Max и устанавливаем счетчик Num в 1, если они равны, то увеличиваем счетчик Num на 1. По окончании цикла переменная Num содержит число повторений максимального элемента массива.</td></tr>
<tr><th colspan="2">Указания по оцениванию</th><th>Баллы</th></tr>
<tr><td colspan="2">Предложен правильный алгоритм, выдающий верное значение (в том числе и алгоритм, требующий двукратного прохода по массиву).
Возможно использование числа 30 вместо константы. Возможно наличие синтаксических ошибок (пропущенные «;», неверная запись оператора присваивания и т.п.), не искажающих замысла автора программы.
В качестве примера правильного и эффективного алгоритма приведен фрагмент программы (заполнение массива опущено):</td><td rowspan="2">2</td></tr>
<tr><td>На языке Паскаль</td><td>На языке Бейсик</td></tr>
</table>

const N = 30;
var a:array[1..N] of integer;
Max, Num, i: integer;
begin
Max := a[1] ;
Num := 1;
for i := 2 to N do
if a[i] > Max then
begin
Max := a[i] ;
Num := 1
end
else
if a[i] = Max then
Num := Num + 1;
writeln(Num)
end.
	N = 30
	DIM i, Max, Num,a(N) AS INTEGER
	Max = a(1)
	Num = 1
	FOR i = 2 TO N
	IF a(i) > Max THEN
	Max = a(i) : Num = 1
	ELSE
	IF a(i) = Max THEN
	Num = Num + 1
	END IF
	END IF
	NEXT i
	PRINT Num
	END	

Имеется не более двух ошибок из числа следующих: 1) Не инициализируется или неверно инициализируются переменные 2) Не указано или неверно указано условие завершения цикла 3) Индексная переменная в цикле не изменяется или изменяется неверно 4) Неверно расставлены операторные скобки		**1**
Ошибок, перечисленных выше, больше двух или алгоритм сформулирован неверно		**0**
Максимальный балл		***2***

Задача С2. Вариант 2.

Текст задачи:

Опишите на русском языке или на одном из языков программирования алгоритм поиска второго по величине (т.е. следующего по величине за максимальным) элемента в числовом массиве из 30 различных элементов.

Ответ (критерии оценивания):

Задание С2 относится к высокому уровню сложности. За решение этого задания можно получить два балла. По содержанию проверяет умение формально описывать известный (изученный в школе) алгоритм на естественном языке или на языке программирования.

<table>
<tr><th colspan="2">Содержание верного ответа и указания по оцениванию</th><th></th></tr>
<tr><td colspan="2">Введем числовые переменные Max1 и Max2, в которых будем хранить соответственно максимальный и следующий за максимальным элемент в уже просмотренной части массива. Первоначально присвоим этим переменным значение первого элемента массива. Затем в цикле до конца массива сравниваем очередной элемент массива с двумя максимальными, и если он больше одного из них или обоих, то меняем два отобранных элемента так, чтобы в Max1 всегда оставался максимальный элемент, а в Max2 следующий по величине . По окончании цикла переменная Max2 содержит второй по величине элемент массива.</td><td></td></tr>
<tr><th colspan="2">Указания по оцениванию</th><th>Баллы</th></tr>
<tr><td colspan="2">Предложен правильный алгоритм, выдающий верное значение (в том числе и алгоритм, требующий двукратного прохода по массиву).
Возможно использование числа 30 вместо константы. Возможно наличие отдельных синтаксических ошибок (пропущенные «;», неверная запись оператора присваивания и т.п.), не искажающих замысла автора программы.
В качестве примера правильного и эффективного алгоритма приведен фрагмент программы:</td><td rowspan="2">2</td></tr>
<tr><td>На языке Паскаль</td><td>На языке Бейсик</td></tr>
</table>

<table>
<tr>
<td>

```
const N=30;
var a:array[1..N] of real;
   Max1, Max2, i: real;
begin
 Max1:=a[1];
 Max2:=a[1];
 if a[2]>Max1 then Max1:=a[2]
          else Max2:=a[2];
 for i:=3 to N do
  begin
   if a[i]>Max1 then
     begin Max2:=Max1;
          Max1:=a[i];
     end
     else if a[i]>Max2 then
          Max2:=a[i];
  end;
  writeln(Max2);
end.
```

</td>
<td>

```
N=30
DIM i, Max1, Max2, a(N) AS REAL
Max1=a(1)
Max2=a(1)
IF a(2)>Max1 THEN Max1=a(2)
ELSE Max2=a(2)
FOR i = 3 TO N
IF a(i)>Max1 THEN
            Max2=Max1
             Max1=a(i)
           ELSE
           IF a(i)>Max2 THEN
              Max2=a(i)
           ENDIF
  ENDIF
NEXT i
PRINT Max2
END
```

</td>
<td></td>
</tr>
<tr>
<td colspan="2">Имеется не более двух ошибок из числа следующих:
1) Не задано или неверно задано первое значение Max1
2) Неверно вычисляется первое значение переменной Max2
3) Не указано условие завершения цикла
4) Индексная переменная в цикле не увеличивается (при использовании циклов while или repeat-until)
5) В программе на Паскале неверно расставлены операторные скобки</td>
<td>1</td>
</tr>
<tr>
<td colspan="2">Ошибок, перечисленных выше, больше двух или алгоритм сформулирован неверно (в частности, не хранится следующий за максимальным элемент).</td>
<td>0</td>
</tr>
<tr>
<td colspan="2">Максимальный балл</td>
<td>2</td>
</tr>
</table>

Рекомендации по оцениванию заданий типа С3

Задача С3. Вариант 1.

Текст задачи:

Два игрока играют в следующую игру. На координатной плоскости стоит фишка. Игроки ходят по очереди. В начале игры фишка находится в точке с координатами (3,2). Ход состоит в том, что игрок перемещает фишку из точки с координатами (x,y) в одну из трех точек: или в точку с координатами (x+3,y), или в точку с координатами (x,y+2), или в точку с координатами (x,y+4). Выигрывает игрок, после хода которого расстояние от фишки до точки с координатами (0,0) больше 12 единиц. Кто выигрывает при безошибочной игре обоих игроков – игрок, делающий первый ход, или игрок, делающий второй ход? Каким должен быть первый ход выигрывающего игрока? Ответ обоснуйте.

Ответ (критерии оценивания):

Содержание верного ответа и указания к оцениванию (допускаются иные формулировки ответа, не искажающие его смысла)

Выигрывает первый игрок, своим первым ходом он должен поставить фишку в точке с координатами (3,4). Для доказательства рассмотрим неполное дерево игры, оформленное в виде таблицы, где в каждой ячейке координаты фишки на каждом этапе игры.

<table>
<tr><th>1 ход</th><th>2 ход</th><th>3 ход</th><th>4 ход</th><th>5 ход</th></tr>
<tr><td>Позиция после первого хода</td><td>II-й игрок (все варианты хода)</td><td>I-й игрок (выигрышный ход)</td><td>II-й игрок (все варианты хода)</td><td>I-й игрок (один из вариантов)</td></tr>
<tr><td rowspan="5"><u>3,4</u></td><td rowspan="3">6,4</td><td rowspan="3"><u>6,6</u></td><td>9,6</td><td><u>12,6</u></td></tr>
<tr><td>6,8</td><td><u>9,8</u></td></tr>
<tr><td>6,10</td><td><u>9,10</u></td></tr>
<tr><td>3,6</td><td><u>6,6</u></td><td colspan="2">Те же варианты 4-5 хода</td></tr>
<tr><td>3,8</td><td><u>3,12</u></td><td colspan="2">Первый игрок выигрывает ответным ходом</td></tr>
</table>

Таблица содержит ***все возможные*** варианты ходов второго игрока. Из неё видно, что при любом ответе второго игрока у первого имеется ход, приводящий к победе.

Указания по оцениванию	Баллы
Правильное указание выигрывающего игрока и его ходов со строгим доказательством правильности (с помощью или без помощи дерева игры).	3
Правильное указание выигрывающего игрока, стратегии игры, приводящей к победе, но при отсутствии доказательства ее правильности.	2
При наличии в представленном решении одного из пунктов: 1. Правильно указан выигрывающий игрок и его первый ход, рассмотрены все возможные ответы второго игрока, но неверно определены дальнейшие действия. 2. Правильно указан выигрывающий игрок и его первый ход, но описание выигрышной стратегии неполно и рассмотрены несколько (больше одного, но не все) вариантов ответов второго игрока.	1
Задание не выполнено или в представленном решении полностью отсутствует описание элементов выигрышной стратегии, и отсутствует анализ вариантов первого-второго ходов играющих (даже при наличии правильного указания выигрывающего игрока).	0
Максимальный балл	3

Задача С3. Вариант 2.

Текст задачи:

Два игрока играют в следующую игру. Перед ними лежат две кучки камней, в первой из которых 5, а во второй – 3 камня. У каждого игрока неограниченно много камней. Игроки ходят по очереди. Ход состоит в том, что игрок или удваивает число камней в какой-то куче, или добавляет 4 камня в какую-то кучу. Выигрывает игрок, после хода которого в одной из куч становится не менее 22 камней. Кто выигрывает при безошибочной игре обоих игроков – игрок, делающий первый ход, или игрок, делающий второй ход? Как должен ходить выигрывающий игрок? Ответ обоснуйте.

Ответ (критерии оценивания):

Содержание верного ответа и указания по оцениванию (допускаются иные формулировки ответа, не искажающие его смысла)
Выигрывает первый игрок. Своим первым ходом он должен удвоить количество камней во второй куче. Для доказательства рассмотрим неполное дерево игры, оформленное в виде таблицы, где в

каждой ячейке записаны пары чисел, разделенные запятой. Эти числа соответствуют количеству камней на каждом этапе игры, в первой и второй кучах соответственно.

	1 ход	2 ход	3 ход	
Стартовая позиция	**I-й игрок (выигрышный ход)**	**II-й игрок (все варианты хода)**	**I-й игрок (выигрышный ход)**	**Пояснение**
5,3	**5,6**	10,6	**10,10**	**Первый игрок выигрывает после любого ответа второго игрока, удвоив число камней в самой большой куче**
		9,6	**9,10**	
		5,10	**9,10** *или* **10,10**	
		5,12	**5,24**	**Выигрыш первого игрока**
	Первые ходы (5,7); (9,3); (10,3) можно не рассматривать, т.к. все они приводят к победе второго игрока при его правильной игре			

Из таблицы видно, что при первом ходе (5,3)->(5,6) первый игрок выигрывает не позже, чем на третьем ходу при любом ответе второго игрока. Дерево игры рассматривает ***все возможные*** варианты.

Указания по оцениванию	***Баллы***
Правильное указание выигрывающего игрока и его ходов со строгим доказательством правильности (с помощью или без помощи дерева игры)	**3**
Правильное указание выигрывающего игрока, стратегии игры, приводящей к победе, но при отсутствии доказательства ее правильности.	**2**
При наличии в представленном решении одного из пунктов: 1. Правильно указан единственно верный ход первого игрока и возможные ответы второго игрока, но неверно (или не полностью) определены дальнейшие действия и	**1**

неправильно указан победитель. 2. Правильно указан выигрывающий игрок, указан его единственно верный первый ход, но описание выигрышной стратегии неполно (например, рассмотрены не все частные случаи ответа второго игрока на первый ход). 3. Имеется полный анализ всех возможных первых ходов обоих играющих, но нет указания на единственно правильный ход первого игрока, и имеются ошибки в определении дальнейшей стратегии игры.	
Задание не выполнено или в представленном решении полностью отсутствует описание элементов выигрышной стратегии, и (даже при наличии правильного указания выигрывающего игрока) отсутствует, либо указание на единственно верный ход первого игрока, либо полный анализ вариантов первых ходов обоих играющих.	**0**
Максимальный балл	***3***

Рекомендации по оцениванию заданий типа С4

Задача С4. Вариант 1.

Текст задачи:

Предположим, что во входном файле `text.dat` содержится текст на английском языке, заканчивающийся точкой (другие символы “.” в этом файле отсутствуют). Требуется написать программу на языке Паскаль или Бейсик, которая будет определять и выводить на экран, какая английская буква встречается в этом тексте чаще всего и сколько именно раз. Строчные и прописные буквы при этом считаются не различимыми. Если таких букв несколько, то программа должна выводить на экран ту из них, которая стоит по алфавиту раньше. Например, пусть файл содержит следующую информацию:

`It is not a simple task. Yes!`

Тогда чаще всего встречаются буквы I, S и T (слово `Yes` в подсчете не участвует, так как расположено после точки). Следовательно, в данном случае программа должна вывести

`I 3`

Ответ (критерии оценивания):

Содержание верного ответа

(допускаются иные формулировки ответа, не искажающие его смысла)

Программа читает текст из файла до точки один раз, подсчитывая в массиве, хранящем 26 целых чисел, количество каждой из букв. Сам текст, при этом, не запоминается. Затем в этом массиве ищется первое вхождение максимального элемента. Баллы начисляются только за программу, которая решает задачу хотя бы для частного случая (например, для строк, состоящих не более чем из 255 символов).

Указания по оцениванию	**Баллы**
Программа работает верно, т.е. определяет первую по алфавиту из наиболее часто встречающихся букв вместе с количеством таких букв, для любых входных данных произвольного размера, просматривает входные данные один раз, не содержит вложенных циклов, в тексте программы не анализируется каждая английская буква в отдельности. Примеры правильных и эффективных программ	4

На языке Паскаль

```
var a:array[ 'A'..'Z'] of integer;
    c, cmax: char;
begin
  assign(input,'text.dat');
  reset(input);
  for c:='A' to 'Z' do a[ c] :=0;
  repeat
    read(c);
    c:= upcase(c);
    if c in [ 'A'..'Z'] then
      a[ c] :=a[ c] +1
  until c='.';
  cmax := 'A';
  for c:= 'B' to 'Z' do
    if a[ c] > a[ cmax] then
      cmax := c;
  writeln(cmax,' ',a[ cmax] )
end.
```

На языке Бейсик

```
DIM i, imax, c, a(26) AS INTEGER
OPEN "TEXT.DAT" FOR INPUT AS #1
S$ = INPUT$(1, #1)
DO WHILE NOT (S$ = ".")
  c = ASC(S$)
  IF(c>=ASC("A")AND c<=ASC("Z")) THEN
   c = c - ASC("A") + 1
  ENDIF
  IF(c>=ASC("a")AND c<=ASC("z")) THEN
   c = c - ASC("a") + 1
  ENDIF
  IF(c>=1 AND c<=26) THEN a(c)=a(c)+1
  S$ = INPUT$(1, #1)
LOOP
imax = 1
FOR i = 2 TO 26
  IF a(i) > a(imax) THEN imax = i
NEXT i
PRINT CHR$(imax + 64), a(imax)
```

	END
Программа составлена верно, но содержит нерациональности: входные данные запоминаются в массиве символов или строке или файл просматривается несколько раз, программа может содержать вложенные циклы. Допускается наличие не более трех синтаксических ошибок.	**3**
Программа составлена в целом верно с вложенными циклами или без, или обрабатывает каждую букву явным образом (26 или 52 оператора IF или оператор CASE, содержащий 26 или 52 вариантов), но, возможно, выводит значение не первой по алфавиту из искомых букв. Возможно в реализации алгоритма содержатся 1–2 ошибки (используется знак “<” вместо “>”, “or” вместо “and” и т.п.). Возможно некорректно организована работа со входным файлом. Допускается наличие не более пяти синтаксических ошибок.	**2**
Программа, возможно, неверно обрабатывает некоторые входные данные, например, отсутствует или предложен некорректный алгоритм обработки строчных или прописных букв или они подсчитываются по отдельности, или программа содержит ошибку в алгоритме поиска максимума. Возможно выводит только искомую букву и не выводит количество букв. Допускается наличие не более семи синтаксических ошибок:.	**1**
Задание не выполнено или выполнено неверно	**0**
Максимальный балл	**4**

Задача С4. Вариант 2.

Текст задачи:

На вход программе подаются 366 строк, которые содержат информацию о среднесуточной температуре всех дней 2004 года. Формат каждой из строк следующий: сначала записана дата в виде dd.mm (на запись номера дня и номера месяца в числовом формате отводится строго два символа, день от месяца отделен точкой), затем через пробел записано значение температуры — число со знаком плюс или минус, с точностью до 1 цифры после десятичной точки. Данная информация отсортирована по значению температуры, то есть хронологический порядок нарушен.

Требуется написать программу на языке Паскаль или Бейсик, которая будет выводить на экран информацию о месяце (месяцах) среднемесячная температура у которого (которых) наименее отклоняется от среднегодовой. В первой строке вывести среднегодовую температуру. Найденные значения для каждого из месяцев следует выводить в отдельной строке в виде: номер месяца, значение среднемесячной температуры, отклонение от среднегодовой температуры.

Ответ (критерии оценивания):

Содержание верного ответа (допускаются иные формулировки ответа, не искажающие его смысла)	
Программа читает входные данные, сразу подсчитывая в массиве, хранящем 12 вещественных чисел, сумму температур в каждом из месяцев, одновременно суммируя все температуры в году. Затем с использованием этого массива ищется минимальное отклонение среднемесячной температуры от среднегодовой. За дополнительный просмотр этого массива распечатывается информация об искомых месяцах. Баллы начисляются только за программу, которая решает задачу хотя бы для частного случая (например, месяц с минимальным отклонением единственен).	
Указания по оцениванию	**Баллы**

Программа работает верно, т.е. определяет все месяцы, в которых среднемесячная температура минимально отклоняется от среднегодовой, не содержит вложенных циклов, в тексте программы не анализируется каждый месяц в отдельности. Допускается наличие в тексте программы одной синтаксической ошибки.
Пример правильной и эффективной программы на языке Паскаль:
`Const d:array[1..12] of integer =`
`(31,29,31,30,31,30,31,31,30,31,30,31);`
`var tm:array[1..12] of real;`
`m:1..12;`
`data:string[5];`
`min,ty,t:real;`
`i:integer;`
`begin`
`for i:=1 to 12 do`
`tm[i]:=0;`
`ty:=0;{среднегодовая температура}`
`for i:=1 to 366 do`
`begin`
`readln(data,t);`
`{вычисляем номер месяца}`
`m:=(ord(data[4])-ord('0'))*10`
`+ord(data[5])-ord('0');`
`tm[m]:=tm[m]+t;`
	4

```
    ty:=ty+t;
  end;
  for i:=1 to 12 do
    tm[i]:=tm[i]/d[i];
  ty:=ty/366;
  min:=100;
  for i:=1 to 12 do
    if abs(tm[i]-ty)<min then
      min:=abs(tm[i]-ty);
  writeln('Среднегодовая температура = ',ty:0:2);
  for i:=1 to 12 do
    if abs(abs(tm[i]-ty)-min)<0.0001 then
      writeln(i,' ',tm[i]:0:2,' ',abs(tm[i]-ty):0:2);
  readln
end.
```

Пример правильной и эффективной программы на языке Бейсик:

```
DATA 31,29,31,30,31,30,31,31,30,31,30,31
DIM i, m, d(12) AS INTEGER
DIM tm(12)
DIM dat AS STRING *5
FOR i = 1 TO 12
    tm(i) = 0
    READ d(i)
NEXT i
ty = 0
'ty – среднегодовая температура
FOR i = 1 TO 366
 INPUT dat, t
'Вычисляем номер месяца
 m=(ASC(MID$(dat,4,1))-ASC("0"))*10+
    ASC(MID$(dat,5,1))-ASC("0")
 tm(m) = tm(m) + t
 ty = ty + t
NEXT i
FOR i = 1 TO 12
  tm(i) = tm(i) / d(i)
```

NEXT i
ty = ty / 366
min = 100
FOR i = 1 TO 12
  IF ABS(tm(i) - ty) < min THEN min = ABS(tm(i) - ty)
NEXT i
PRINT "Среднегодовая температура = ";
PRINT USING "##.##"; ty
FOR i = 1 TO 12
  IF ABS(ABS(tm(i) - ty) - min) < .0001 THEN
  PRINT i;
  PRINT USING "##.## "; tm(i); ABS(tm(i) - ty)
  END IF
NEXT i
END	

Программа составлена верно, но содержит вложенные циклы (от 1 до 12 и от 1 до 366), возможно, запоминает значения всех температур в массиве. Допускается наличие от одной до трех синтаксических ошибок.	**3**
Программа составлена в целом верно с вложенными циклами или без них, или обрабатывает каждый месяц явным образом (12 операторов IF или оператор CASE, содержащий 12 вариантов), или предварительно сортирует входные данные в хронологическом порядке. Возможно, выводит значение только одного месяца с минимальным отклонением температуры. Возможно в реализации алгоритма содержатся 1–2 ошибки (используется знак “<” вместо “>”, “or” вместо “and” и т.п.). Допускается наличие не более пяти синтаксических ошибок.	**2**
Программа, возможно, некорректно определяет номер месяца или неверно вводит или выводит данные, или содержит ошибку в алгоритме поиска минимума или средней температуры, или отклонение берется не по абсолютной величине. Допускается наличие не более семи синтаксических ошибок.	**1**
Задание не выполнено или выполнено неверно	**0**
Максимальный балл	**4**

Инструкции и памятки по проверке работ

Задача С1

Вариант 1.

1. Прежде всего следует убедиться, что учащийся привел пример такой точки (пары чисел x и y), для которой исходная программа НЕВЕРНО решает поставленную задачу. Какого-либо обоснования данного элемента ответа от учащегося не требуется и за отсутствие обоснования оценка не снижается.
2. На основании критериев оценить предложенную учащимся доработку программы. Для этого нужно убедиться, что

А) исправления учащегося действительно устраняют ошибки в исходном тексте, указанные в критериях;

Б) при внесении исправлений учащийся не допустил новых логических ошибок.

Если программа исправлена учащимся не до конца, т.е. остаются ситуации, в которых она неверно решает поставленную задачу, то следует оценить, работает ли правильно программа в большем числе случаев, чем исходная, т.е. улучшила ли доработка исходную программу.

Вариант 2.

1. Проверка также многошаговая. Необходимо подобрать значения a, b, x так, **b > a**. Обоснования к ответу не нужно приводить (см. п. 2 Описание сложных для оценивания ситуаций).
2. Необходимая доработка программы должна обязательно учитывать оба условия, что возможно a>b и a<b. Возможная доработка:

```
if a<b then begin p:=a; a:=b; b:=p end;
 if (a>x) AND (x>b) then
writeln(' x между a,b');
```

(могут быть и другие правильные способы доработки)

3. Если осуществить доработку без использования логических связок, то пункт 2 можно опустить, поскольку это и будет являться наиболее эффективным решением. Возможная доработка без использования связок AND, OR:

```
p:=(x-a)*(x-b);  if p<0 then
```

writeln(' x между a,b');

(могут быть и другие способы доработки без использования связок AND, OR)

4. При оценке других вариантов доработки программы нужно проверять, что поставленная цель достигается

Задача С2

- Определите, на каком языке записан алгоритм и при необходимости наведите справки о синтаксисе избранного абитуриентом языка программирования. В случае, если язык экзотический и Вы его не знаете, попытайтесь оценить работу из общих соображений.
- Сравните описание алгоритма с имеющимися образцами. В случае совпадения - оцените в соответствии с рекомендациями.
- Если описание алгоритма не совпадает с образцами, а ошибки в описании алгоритма с первого взгляда не видны, осуществите формальное исполнение алгоритма с тестовыми примерами исходных данных. Длину тестового массива следует сократить до 4-6 элементов. При тестах необходимо особенно тщательно проверять «критические» случаи, например, когда элементы массива одинаковы или изначально упорядочены. Оцените правильность полученных результатов.
- В случае алгоритма, представленного на русском языке или в виде-блок-схемы необходимо оценить возможность выполнения это алгоритма человеком и уровень детализации алгоритма (должны выполняться требования дискретности, детерминированности и результативности) .
- При оценке алгоритма отметьте все ошибки, упомянутые в критериях оценивания. В случае, если таких ошибок более двух, сразу снижайте оценку до 0 баллов.
- Не допускайте произвольного ужесточения критериев оценивания. Не вводите дополнительных ограничений. Не оценивайте «стиль» программирования, аккуратность записи, наличие отступов и прочие важные, но данным заданием не проверяемые вещи.
- Не забывайте, что эффективность алгоритмов в данной задаче **не оценивается, поэтому не следует снижать оценку за решение, в котором используется два просмотра массива**.

Задача С3

- Ознакомтесь со стратегией решения имеющегося варианта задачи С3, которое представлено в материалах для эксперта (Раздел 1.4 данного документа).

- Решение может быть оформлено по-разному: в виде таблиц, графов, словесного описания, и т.д. Стратегия правильного решения в работе экзаменуемого может быть представлена в варианте отличном от «образцового», но основные положения должны совпадать с предложенным.

- Важно вычленить в решении наличие/отсутствие осмысленной стратегии играющих, отбор/отбраковки ходов приводящих к победе/поражению, полноты рассмотенных вариантов.

- Проверьте, правильно ли назван игрок, побеждающий при безошибочной игре? Есть ли обоснование этого факта?

- Верное указание выигрывающего игрока, не обоснованное ничем разумным, оценивается как неправильное решение в «0» баллов.

- Проверьте, есть ли описание выигрышной стратегии? Полно ли это описание? Нет ли каких-либо упущений?

- Рассмотрены ли все возможные варианты ходов? Для очевидных проигрышных вариантов достаточно упоминания этого факта, но оно должно быть.

- Возможны работы, в которых проведен подробный анализ вариантов хода первого игрока и возможных ответов второго, но в дальнейших рассуждениях допущена ошибка и победитель указан неверно. Тем не менее, такое решение можно оценить в «1» балл.

- Руководствуйтесь рекомендациями по оцениванию заданий типа С3 (Раздел 1.4 данного документа).

Задача С4

- Определите язык программирования, на котором написана программа. Программы, написанные на языках программирования, отличных от Бейсика и Паскаля, в частности на языке Си, тоже должны быть оценены. Это же относится и к современным версиями языков Паскаль (Дельфи) и Бейсик (Вижуал-Бейсик). При необходимости эксперт может воспользоваться справочной литературой.

- Подсчитайте число синтаксических ошибок в данной программе.
- Рассмотрите реализацию каждой части алгоритма. Определите количество алгоритмических ошибок и количество ошибок в реализации алгоритмов.
- Если ошибок мало и программа оценивается из 3-4 баллов, то оцените эффективность предложенного решения.
- Оцените решение задачи согласно критериям оценивания 1.5.

Описание ситуаций сложных для оценивания

Задача С1

Вариант 1.

Первый элемент ответа (оценивается из одного балла)

Ситуация	Пояснение	Оценка за элемент ответа
Приведен пример исходных данных, при которых программа работает **правильно** (с обоснованием или без)	Условие задачи не выполнено.	0
Приведен пример исходных данных, при которых программа работает **неправильно,** но без какого-либо обоснования.	Условие задачи выполнено.	1
В качестве решения приведен график из условия задачи с указанием точки (точек), при которых программа работает **неправильно**	Условие задачи выполнено. Исходные данные можно указать и как точку на плоскости.	1

Второй элемент ответа (оценивается из двух баллов)

Ситуация	Пояснение	Оценка за элемент ответа
Программа написана учащимся заново, возможно, с использованием принципиально другого способа решения и на языке программирования, отличном от приведенных в условии	Программу следует проверить и оценить в соответствии с критериями.Если она работает верно для всех исходных данных – 2 балла*. Если она работает верно для тех же данных, что исходная, и ещё	0-2

	для некоторого множества данных – 1 балл. В остальных случаях – 0 баллов. ***Примечание. Здесь и далее в таблице для краткости предполагается, что доработанная программа не содержит синтаксических ошибок. При их наличии балл снижается в соответствии с критериями.**	
Учащийся привел решение формально верное, но нерациональное и неэффективное	Условие задачи выполнено.	2
Доработка программы предложена в виде фраз на естественном языке, а не на языке программирования.	Условие задачи не выполнено.	0
Доработанная программа стала верно работать на довольно большом наборе данных, но перестала верно работать на всех или на части данных, на которых работала верно исходная программа.	Условие задачи не выполнено.	0

Вариант 2.

Вопрос 1 к заданию подразумевает проверку понимания условия задания. Это означает понимание смысла используемых логических операций и, что **a** может быть больше **b**, либо наоборот.

Возможны следующие случаи при оценивании:

Ответ	Обоснование	Оценивание
Указаны варианты значений с соблюдением условия b>a	Отсутствует	1 балл
Указаны варианты значений с соблюдением условия b>a	Приведено, типа: «Для правильной работы программы должно соблюдаться условие (a>x) AND (x>b), для неправильной наоборот» или «Программой не предусмотрен данный случай»	1 балл
Указаны варианты значений для обратного условия b<a	Не имеет значения	0 баллов
Указаны варианты отрицательных значений хотя бы одного из чисел a, b, x при любых условиях.	Не имеет значения.	0 баллов

Вопрос 2 к заданию является продолжением первого, подключается умение реализовать на языке программирования приведенное в задании условие.

Возможны следующие случаи при оценивании:

Ответ	Обоснование	Оценивание
Варианты доработки учитывающие оба условия: что возможно a>b и a<b.	Не имеет значения	1 балл
Варианты доработки, учитывающие только одно условие.	Не имеет значения	0 баллов

Вопрос 3 к заданию проверяет знание эквивалентов логических операций и умение реализовать их в программе.

Возможны следующие случаи при оценивании:

Ответ	Обоснование	Оценивание
Варианты доработки учитывают оба	Не имеет значения	2 балла

условия: что, возможно, a>b и a<b, а также не содержат логических операций.		
Варианты доработки учитывают оба условия: что, возможно, a>b и a<b, а также не содержат логических операций, но содержат более двух ошибок.	Не имеет значения	1 балл
Варианты доработки, учитывающие только одно условие.	Не имеет значения	0 баллов

Задача С2

Основная сложность при оценивании задач С2 состоит в том, что задание не устанавливает четко, каким образом алгоритм обработки массива может быть записан. Разрешение использовать естественный язык автоматически приводит к употреблению абитуриентами неточных и расплывчатых формулировок, оборотов «аналогично ищем максимальный элемент», «производим подсчет этих чисел и выводим результат» и т.д. При оценке алгоритмов, записанных на естественном языке, основным критерием должна быть возможность их формального исполнения. То есть описание должно быть максимально приближено к записи команд, соответствующих основным операторам языков программирования: должно быть указано какие переменные заводятся, как выполняется присваивание, каково условие завершения цикла и так далее. При этом, например, фраза «Для каждого элемента массива мы выполняем сравнение его с переменной…» может рассматриваться как вполне строгое описание цикла с параметром.

При рассмотрении записей алгоритмов на естественном языке следует очень внимательно проследить, как абитуриент описывает операцию присваивания. Наряду с формулой «Присваиваем переменной *m* значение очередного элемента массива» может употребляться и конструкция «присваиваем значение очередного элемента массива переменной *m*» (часто употребляется конструкция «записываем в …»), а то и «присваиваем *i* эл. массива значение максимального» (на самом деле говорится все о той же операции $m:=A[i]$). Видимо, не стоит снижать оценку за косноязычие, если из описания алгоритма видно, что абитуриент четко представляет себе его выполнение. Если же алгоритм не может быть формально исполнен без каких-то особых

разъяснений (формально можно описать это как вызов процедуры или вспомогательного алгоритма), то такой алгоритм должен быть оценен нулем баллов. Пример такого описания: «Сначала находим максимальное значение массива, а затем…».

Так же следует оценивать применение в программах на языке программирования стандартных функций из библиотек (сортировка массива по возрастанию, например): задание проверяет не знание названий стандартных функций, а знание алгоритмов.

Формулировка задания не предполагает возможности записи алгоритма в виде блок-схемы, но на практике абитуриенты довольно часто рисуют блок-схему алгоритма вместо его описания на формальном или естественном языке. Блок-схемы должны оцениваться по тем же критериям, что и алгоритмы, записанные на языке программирования. При этом оценка за синтаксические ошибки в блок-схемах (например, запись условия ветвления в блоке-прямоугольнике вместо ромба) не снижается.

Еще одной сложностью при проверке этого задания является разнообразие языков программирования, на которых записываются решения: от классического Бейсика с метками и оператором GOTO до C++, PHP и Perl. Это обстоятельство не должно смущать, так как простота формулировки задания предполагает и простую его реализацию на языке программирования, использование небольшого набора базовых операторов, достаточно инвариантных. Задачей экзаменатора при анализе решения **этой задачи (С2)** не является проверка знания абитуриентом синтаксиса избранного языка, а только знание им (ею) алгоритмов и умения записать их с помощью средств формализации.

Таким образом, за синтаксические ошибки в решениях заданий С2 оценка не снижается.

Некоторые языки (Паскаль) требуют точного описания типов переменных, некоторые осуществляют автоматическое определение и преобразование типов. Исходя из соображений равенства условий экзамена для всех абитуриентов ошибки в описании типов переменных на тех языках, где это требуется, **не ведут** к снижению оценки.

Большинство экзаменуемых пишет программы, предусматривающие двойной проход по массиву. Такие программы, если они написаны без ошибок, оцениваются **двумя баллами**.

Вариант 1.

В том случае, если задача решается за два прохода по массиву, она становится совсем несложной. Типичная ошибка – неверное задание первого значения переменной *Max* (чаще всего указывается 0). В том случае, если в массиве нет положительных элементов, алгоритм работать не будет. В случае подобной ошибки оценка снижается на один балл.

Первое значение переменной *Num* перед вторым проходом должно быть 0.

При решении задачи в один проход по массиву при нахождении очередного локального максимума часто не обновляется переменная *Num*. За такую ошибку должна быть выставлена оценка 0, так как алгоритм не будет работать во всех случаях, а не только в некоторых, как при неверном задании первого значения переменной *Max.*

При анализе алгоритмов, записанных на естественном языке, следует особое внимание уделить полноте и формальной точности описания операций.. Очень часто экзаменуемые отделываются общими словами. Фраза «Находим максимальные элементы массива и подсчитываем их число» должна быть оценена как недостаточная для формального исполнения, за решение в этом случае должно быть выставлено 0 баллов.

Вариант 2.

Задание почти всегда решается экзаменуемыми в два прохода. Следует обратить особое внимание на задание первого значения переменной *Max2*. Также следует обратить внимание на проверку условия: изменение переменной *Max2* происходит только в случае выполнения неравенства $Max2 < A[i] < Max1$. Экзаменуемый должен либо воспользоваться логической функцией AND либо двумя вложенными условными операторами. Следует особо проследить за правильностью расстановки операторных скобок.

При анализе алгоритмов, записанных на естественном языке, следует особое внимание уделить полноте и формальной точности описания операций, осуществляемых при втором проходе массива. Фактически операция поиска максимума повторяется снова, но при этом каждый раз проверяется, что найденный локальный максимум меньше найденного максимального значения для всего массива. Часто экзаменуемые предлагают решения, работающие только в ряде случаев (если второй по значению элемент имеет индекс больший, чем максимальный, или наоборот). В случае

затруднения при оценке решения лучше всего попытаться исполнить предложенный алгоритм на разных наборах исходных данных.

При анализе алгоритмов, записанных на естественном языке, необходимо обращать внимание на то, описаны ли первые значения переменных *Max1* и *Max2*. В случае, если этого не сделано, оценка не может быть выше 1 балла.

Задача С3

Вариант 1.

Не стоит начинать оценивание работы с проверки правильности указания выигрывающего игрока. Анализ решения удобно начинать с просмотра дерева игры. Если дерево представлено наглядно, в том, или ином графическом варианте, то проверить логичность размышлений, а также их полноту несложно.

Более трудным для оценивания является случай, когда представлено решение, не содержащее дерева игры в явном виде. В этом случае необходимо последовательно проверить:

- наличие описания стратегии ходов играющих;
- полнообъемность рассмотрения вариантов ходов, т.е. нет ли в представленном решении потерь какого-либо из возможных шагов игравших, особенно, принципиально меняющих характер развития событий;
- что высказывание о правильном указании на выигрывающего игрока действительно базируется на логически обоснованном выводе.

Один балл может быть снят в случае, если при наличии правильного описания самой выигрышной стратегии, нет указания на полноту предъявленного рассмотрения (в явном виде, или в виде простого полного перебора возможных вариантов и т.д.), т.е. отсутствует доказательство правильности выигрышной стратегии.

Применение указаний по оцениванию дает возможность оценить работы экзаменуемых.

Вариант 2.

Анализ решения удобно начинать с просмотра дерева игры. Если дерево представлено наглядно, в том, или ином графическом варианте, то проверить логичность размышлений, а также их полноту несложно.

Более трудным для оценивания является случай, когда представлено решение, не содержащее дерева игры в явном виде. В этом случае необходимо последовательно проверить:

- наличие описания стратегии ходов играющих;
- полнообъемность рассмотрения вариантов ходов, т.е. нет ли в представленном решении потерь какого-либо из возможных шагов игравших, особенно, принципиально меняющих характер развития событий;
- что высказывание о правильном указании на выигрывающего игрока действительно базируется на логически обоснованном выводе.

Правильное определение выигрывающего игрока важно, но много важнее наличие у экзаменуемого умения правильно анализировать стратегию, рассматривать множество вариантов развития ситуации.

Один балл может быть снят в случае, если при наличии правильного описания самой выигрышной стратегии, нет указания на полноту предъявленного рассмотрения (в явном виде, или в виде полного перебора возможных вариантов и т.д.), т.е. отсутствует доказательство правильности выигрышной стратегии.

Применение указаний по оцениванию дает возможность оценить работы экзаменуемых.

Задача С4

Вариант 1.

1) Экзаменуемый не умеет работать с файлами и считывает данные с клавиатуры. Это расценивается как ошибка считывания входных данных из файла и работа оценивается из двух баллов.

2) В решение присутствует опечатка, в результате которой оно становится синтаксически или логически неверным (a[i] вместо a[j], a[i] вместо a[i].x и т.п.). Если в результате опечатки оценка решения согласно критериям должна быть снижена до 1–2 баллов, то ее можно рассматривать как синтаксическую ошибку и оценивать решение из 3 баллов.

3) Технические программистские ошибки (например, отведение недостаточного количества памяти, не обнуление значения переменной или выход за границу массива, некорректное использование стандартных функция соответствующего языка

программирования) следует приравнивать к синтаксическим, если они не порождены ошибками в алгоритме решения.

4) По невнимательности решается не совсем та задача. Например, выводится сколько каких букв встречается в тексте, а не ищется наиболее часто встречаемая буква. Те части решения, которые в программе отсутствуют (например, поиск максимума) следует приравнивать к ошибкам в соответствующих алгоритмах.

5) Программа написана на языках программирования, отличном от Бейсика или Паскаля, в частности на языке Си. Такая программа тем не менее должна быть оценена. Это же относится и к особенностям современных версий языков Паскаль (Дельфи-Паскаль) и Бейсик (Вижуал-Бейсик). При необходимости эксперт может воспользоваться справочной литературой.

Вариант 2.

1) Экзаменуемый неверно организует считывание данных, в результате чего программа оказывается неработоспособной. В данном случае можно предположить, что при отладке программы в среде программирования, данная ошибка была бы исправлена, поэтому ее надо рассматривать как одну из ошибок, а основную часть программы оценивать в предположении, что данные введены корректно.

2) В решение присутствует опечатка, в результате которой оно становится синтаксически или логически неверным (a[i] вместо a[j], a[i] вместо a[i].x и т.п.). Если в результате опечатки оценка решения согласно критериям должна быть снижена до 1–2 баллов, то ее можно рассматривать как синтаксическую ошибку и оценивать решение из 3 баллов.

3) Технические программистские ошибки (например, отведение недостаточного количества памяти, не обнуление значения переменной или выход за границу массива, некорректное использование стандартных функция соответствующего языка программирования) следует приравнивать к синтаксическим, если они не порождены ошибками в алгоритме решения.

4) По невнимательности решается не совсем та задача. Например, для каждого месяца выводится значение отклонения среднемесячной температуры от среднегодовой, а не ищется минимальное отклонение. Те части решения, которые в программе отсутствуют (например, поиск минимума) следует приравнивать к ошибкам в соответствующих алгоритмах, если об этом явно не говорится в критериях оценивания.

5) Программа написана на языках программирования, отличном от Бейсика или Паскаля, в частности на языке Си. Такая программа тем не менее тоже должна быть оценена. Это же относится и к особенностям современных версий языков Паскаль (Дельфи-Паскаль) и Бейсик (Вижуал-Бейсик). При необходимости эксперт может воспользоваться справочной литературой.