МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Утверждаю
Первый проректор –
проректор по учебной работе
_____Е. Е. Чупандина

Дополнительная образовательная программа
повышения квалификации

«Подготовка экспертов для работы в региональной предметной комиссии при проведении государственного итоговой аттестации по образовательным программам среднего и общего образования по предмету «Информатика и ИКТ»

Категория обучающихся учителя средних общеобразовательных организаций, преподаватели высших учебных заведений

Объем программы 24 (час.)

Форма обучения очная

Воронеж

I. Общая характеристика программы

1.1. Цели реализации программы

- Для программ повышения квалификации:

Совершенствование компетенции, необходимой для профессиональной деятельности в качестве эксперта предметной комиссии по проверке ответов заданий с развернутым ответом участников единого государственного экзамена (ЕГЭ) по информатике и ИКТ.

1.2. Планируемые результаты обучения

- Для программ повышения квалификации:

– Владение содержанием контрольно-измерительных материалов единого государственного экзамена (ЕГЭ) по информатике и ИКТ;

– Знание критериев оценки ответов участников единого государственного экзамена (ЕГЭ) по информатике и ИКТ и навыки их применения при работе в качестве экспертов предметной комиссии по информатике и ИКТ.

II. Учебный план

	Наименование разделов и дисциплин	Всего, час.	В том числе			Форма контроля
			лекции	практические и лабораторные занятия	самостоятельная работа	
1.	Структура КИМов единого государственного экзамена по информатике и ИКТ	2	2		2	Опрос
2.	Характеристика заданий первой части КИМов единого государственного экзамена по информатике и ИКТ	6	4	2	2	Собеседование
3.	Характеристика заданий второй части КИМов единого государственного экзамена по информатике и ИКТ	8	4	4	2	Опрос
4.	Критерии оценки заданий второй части КИМов единого государственного экзамена по информатике и ИКТ	6	4	2	2	Опрос

5.	Итоговая аттестация	2		2		Зачет
6.	Итого	24	14	10	8	

Руководитель дополнительной образовательной программы _____ Н. А. Каплиева

III. Рабочая программа

Цели курса: Совершенствование компетенции, необходимой для профессиональной деятельности в качестве эксперта предметной комиссии по проверке ответов заданий с развернутым ответом участников единого государственного экзамена (ЕГЭ) по информатике и ИКТ.

Задачи курса:

Формирование знаний о содержании контрольно-измерительных материалов единого государственного экзамена (ЕГЭ) по информатике и ИКТ;

Освоение критериев оценки ответов участников единого государственного экзамена (ЕГЭ) по информатике и ИКТ и развитие навыков их применения при работе в качестве экспертов предметной комиссии по информатике и ИКТ.

1. **Компетенции обучающегося**, формируемые в результате освоения программы:

– *Владение содержанием КИМов единого государственного экзамена по информатике и ИКТ*

– *Владение критериями оценивания ответов участников единого государственного экзамена по информатике и ИКТ*

– *Владение навыками использования критериев оценивания при работе в качестве эксперта предметной комиссии по информатике и ИКТ*

2. **Содержание разделов**

№ п/п	Наименование раздела	Содержание раздела дисциплины
1	Структура КИМов единого государственного экзамена по информатике и ИКТ (2 час.)	Структура заданий части 1 с кратким ответом. Структура заданий части 2 с ответом в развернутой форме
2	Характеристика заданий первой части КИМов единого государственного экзамена по информатике и ИКТ	Задания с кратким ответом в виде числа или последовательности символов. Информация и ее кодирование. Системы, компоненты, состояние и взаимодействие компонентов. Моделирование. Системы счисления. Логика и алгоритмы
3	Характеристика заданий второй части КИМов единого государственного экзамена по информатике и ИКТ	Распределение заданий с развернутым ответом по уровню сложности. Алгоритмы обработки массивов, последовательностей. Алгоритмы обработки записи натурального числа в позиционной системе. Построение дерева игры по заданному алгоритму и обоснование выигрышной стратегии
4	Критерии оценки заданий второй части КИМов единого государственного экзамена по информатике и ИКТ	Система оценивания выполнения заданий с развернутым ответом и экзаменационной работы в целом. Структура протокола проверки развернутых ответов

3. Методические рекомендации и пособия по реализации учебной программы.

Материалы ФИПИ. Методические материалы для председателей и членов предметных комиссий субъектов Российской Федерации по проверке выполнения заданий с развернутым ответом экзаменационных работ ЕГЭ 2018 года. ИНФОРМАТИКА и ИКТ. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ОЦЕНИВАНИЮ ВЫПОЛНЕНИЯ ЗАДАНИЙ ЕГЭ С РАЗВЕРНУТЫМ ОТВЕТОМ

4. Контрольные задания

Тренинг по оцениванию задания 24.

Задание 24 проверяет умение анализировать текст программы с точки зрения соответствия записанного алгоритма поставленной задаче и изменять его в соответствии с заданием.

Разберите варианты решений, представленных учащимися, и проведите процедуру оценивания в соответствии с критериями.

24 Требовалось написать программу, при выполнении которой с клавиатуры считывается натуральное число N, не превосходящее 10^9, и выводится количество цифр этого числа. Программист торопился и написал программу неправильно. (Ниже для Вашего удобства программа представлена на четырёх языках программирования.)

Бейсик

```
DIM N AS LONG
INPUT N
sum = 0
WHILE N >= 9
 N = N \ 10
 sum = sum + 1
WEND
PRINT sum
END
```

Паскаль

```
var N: longint;
 sum: integer;
begin
 readln(N);
 sum := 0;
 while N >= 9 do
 begin
 N := N div 10;
 sum := sum + 1;
 end;
 writeln(sum);
end.
```

Си

```
#include<stdio.h>
int main()
{
 long int N;
 int sum;
 scanf("%ld", &N);
 sum = 0;
 while (N >= 9)
 {
 N = N / 10;
 sum = sum + 1;
 }
 printf("%d", sum);
}
```

Алгоритмический язык

```
алг
нач
 цел N, sum
 ввод N
 sum := 0
 нц пока N >= 9
 N := div(N, 10)
 sum := sum + 1
 кц
 вывод sum
кон
```

Последовательно выполните следующее.

1. Напишите, что выведет эта программа при вводе числа 584.

2. Укажите число, для которого программа будет работать верно.

3. Найдите все ошибки в этой программе (их может быть одна или несколько). Укажите все строки (одну или более), содержащие ошибки, и для каждой такой строки приведите правильный вариант.
Обратите внимание, что требуется найти ошибки в имеющейся программе, а не написать свою, возможно, использующую другой алгоритм решения. Исправление ошибки должно затрагивать только строку, в которой находится ошибка.

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

1. Программа выведет число 2.
2. Программа работает верно для всех чисел, начинающихся на 9, в том числе для числа 9. [Достаточно указать любое такое число.]
3. В качестве ответа для остальных чисел программа выдаёт число на 1 меньшее, чем нужно. Возможные (не все) варианты исправления для языка Паскаль:
1) исправление условия продолжения цикла на
while (N >= 1) do или while (N > 0) do
При этом замена на
while (N >= 0) do
корректной не является.
2) исправление инициализации на
sum := 1
а условие продолжения цикла на
while (N > 9) do или while (N >= 10) do	

Указания по оцениванию	**Баллы**
Обратите внимание! В задаче требовалось выполнить **три** действия. Баллы за данное задание начисляются как сумма баллов за верное выполнение каждого действия. 1. Верно указано, что именно выведет программа при указанных в условии входных данных. 2. Указано число, при котором программа работает верно. 3. Указаны все строки (одна или более), в которые нужно внести исправления, и эти исправления внесены; при этом получена верно работающая программа. При выполнении действия 3 верное указание на ошибку при неверном её исправлении не засчитывается. Обратите внимание! Выбор ошибочных строк может быть выполнен не единственным способом. В работе (во фрагментах программ) допускается наличие отдельных синтаксических ошибок, не искажающих замысла автора решения	
Правильно выполнены все три действия	3
Правильно выполнены два действия из трёх	2
Не выполнены условия, позволяющие поставить 2 или 3 балла, однако выполнено одно из следующих условий. 1. Выполнено одно действие из трёх. 2. Представлен новый верный текст программы, возможно, совершенно не похожий на исходный	1
Все пункты задания выполнены неверно или отсутствуют	0
Максимальный балл	*3*

Тренинг по оцениванию задания 25.

Задание 25 проверяет умение написать короткую (10–15 строк) простую программу обработки массива на языке программирования.

Разберите варианты решений, представленных учащимися, и проведите процедуру оценивания в соответствии с критериями.

25 Дан целочисленный массив из 30 элементов. Элементы массива могут принимать значения от 150 до 200 – рост учащихся выпускного класса. В команду по автогонкам

входят все учащиеся, чей рост не более 175 см. Гарантируется, что такие учащиеся в классе есть. Запишите на одном из языков программирования алгоритм, который находит и выводит рост самого высокого участника гоночной команды.

Исходные данные объявлены так, как показано ниже. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать часть из них.

Бейсик

```
N=30
DIM A(N) AS INTEGER
DIM I, J, MAX AS INTEGER
FOR I = 1 TO N
   INPUT A(I)
NEXT I
 ...
END
```

Паскаль

```
const
 N=30;
var
 a: array [1..N] of integer;
 i, j, max: integer;
begin
    for i:=1 to N do
readln(a[i]);
...
end.
```

Си

```
#include <stdio.h>
#define N 30
void main(void)
{
int a[N];
int i, j, max;
for (i=0; i<N; i++)
    scanf("% d", &a[i]);
...
}
```

Алгоритмический язык

```
алг
нач
 цел N = 30
 целтаб a[1:N]
 цел i, j, max
 нц для i от 1 до N
    ввод a[i]
 кц
 ...
кон
```

В качестве ответа необходимо привести фрагмент программы, который должен находиться на месте многоточия. Вы можете записать решение также на другом языке программирования (укажите название и используемую версию языка программирования, например, Borland Pascal 7.0). В этом случае вы должны использовать те же самые исходные данные и переменные, какие были предложены в условии (например, в образце, записанном на Алгоритмическом языке).

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

На языке Паскаль

```
max:=150;
for i:=1 to N do
    if (a[i]<=175) and (a[i]>max)
then
        max:=a[i];
writeln(max);
```

На языке Бейсик

```
MAX = 150
FOR I = 1 TO N
   IF A(I) <= 175 AND A(I) > MAX
THEN
         MAX = A(I)
    ENDIF
NEXT I
PRINT MAX
```

На языке С

```
max=150;
for (i=0; i<N; i++)
     if (a[i]<=175 && a[i]>max)
         max=a[i];
printf("% d", max);
```

На Алгоритмическом языке

```
max = 150
нц для i от 1 до N
     если a[i]<=175 и a[i]> max
        то max := a[i]
     все
кц
вывод max
```

Указания по оцениванию	Баллы
Предложен правильный алгоритм, выдающий верное значение. Допускается запись алгоритма на другом языке, использующая аналогичные переменные. В случае, если язык программирования использует типизированные переменные, описания переменных должны быть аналогичны описаниям переменных на Алгоритмическом языке. Использование нетипизированных или необъявленных переменных возможно только в случае, если это допускается языком программирования, при этом количество переменных и их идентификаторы должны соответствовать условию задачи. В программе допускается наличие отдельных синтаксических ошибок, не искажающих замысла автора программы.	2
В любом варианте решения может присутствовать не более одной ошибки из числа следующих. 1. Не инициализируется или неверно инициализируется переменная MAX (например, ей присваивается значение a[1] или число большее 150). 2. В сравнении со 175 вместо знака «меньше или равно» используется знак «меньше». 3. Отсутствует вывод ответа. 4. Используется переменная, не объявленная в разделе описания переменных. 5. Не указано или неверно указано условие завершения цикла. 6. Индексная переменная в цикле не меняется (например, в цикле while). 7. Неверно расставлены операторные скобки.	1
Ошибок, перечисленных в п. 1–7, две или больше, или алгоритм сформулирован неверно.	0
Максимальный балл	2

Тренинг по оцениванию задания 26.

Задание 26 проверяет умение построить дерево игры по заданному алгоритму и обосновать выигрышную стратегию.

Разберите варианты решений, представленных учащимися, и проведите процедуру оценивания в соответствии с критериями.

26 Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежит куча камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в кучу один камень или увеличить количество камней в куче в два раза. Например, имея кучу из 15 камней, за один ход можно получить кучу из 16 или 30 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней.
Игра завершается в тот момент, когда количество камней в куче становится не менее 22. Победителем считается игрок, сделавший последний ход, то есть первым получивший кучу, в которой будет 22 или больше камней.
В начальный момент в куче было S камней, $1 \le S \le 21$.

> Говорят, что игрок имеет *выигрышную стратегию*, если он может выиграть при любых ходах противника. Описать стратегию игрока – значит описать, какой ход он должен сделать в любой ситуации, которая ему может встретиться при различной игре противника.

Выполните следующие задания. Во всех случаях обосновывайте свой ответ.

1. а) При каких значениях числа S Петя может выиграть первым ходом? Укажите все такие значения.
   б) Укажите такое значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом. Опишите выигрышную стратегию Вани.
2. Укажите два значения S, при которых у Пети есть выигрышная стратегия, причем (а) Петя не может выиграть первым ходом, но (б) Петя может выиграть своим вторым ходом, независимо от того, как будет ходить Ваня.
   Для указанных значений S опишите выигрышную стратегию Пети.

3. Укажите такое значение *S*, при котором у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети, но при этом у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.

Для указанного значения *S* опишите выигрышную стратегию Вани. Постройте дерево всех партий, возможных при этой выигрышной стратегии Вани (в виде рисунка или таблицы). На ребрах дерева указывайте, кто делает ход, в узлах – количество камней в позиции.

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

(допускаются иные формулировки ответа, не искажающие его смысла)

1. а) Петя может выиграть первым ходом, если *S* =11, …, 21. Во всех случаях нужно удвоить количество камней в куче. При меньших значениях *S* за один ход нельзя получить кучу, в которой больше 21 камня.
   б) Ваня может выиграть первым ходом (как бы ни играл Петя), если исходно в куче будет *S* =10 камней. Тогда после первого хода Пети в куче будет 11 камней или 20 камней. В обоих случаях Ваня удваивает количество камней и выигрывает первым ходом.

2. Возможные значения *S*: 5 и 9. В этих случаях Петя, очевидно, не может выиграть первым ходом. Однако он может получить кучу из 10 камней. Эта позиция разобрана в п. 1б. В ней игрок, который будет ходить (теперь это Ваня), выиграть не может, а его противник (то есть, Петя) следующим ходом выиграет.
3. Возможное значение *S*: 8. После первого хода Пети в куче будет 9 или 16 камней. Если в куче станет 16 камней, Ваня удвоит количество камней и выиграет первым ходом. Ситуация, когда в куче 9 камней, разобрана в п. 2. В этой ситуации игрок, который будет ходить (теперь это Ваня), выигрывает своим вторым ходом.
   В таблице изображено дерево возможных партий при описанной стратегии Вани. Заключительные позиции (в них выигрывает Ваня) подчеркнуты. На рисунке это же дерево изображено в графическом виде (оба способа изображения дерева допустимы).

	Положения после очередных ходов			
И.п.	1-й ход Пети (разобраны все ходы)	1-й ход Вани (только ход по стратегии)	2-й ход Пети (разобраны все ходы)	2-й ход Вани (только ход по стратегии)
8	8+1 =**9**	9+1=**10**	10+1=**11**	11*2=**22**
			10*2=**20**	20*2=**40**
	8*2=**16**	16*2=**32**		

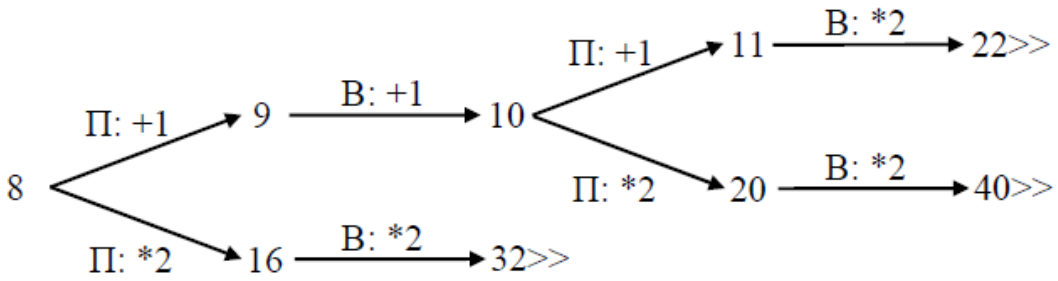

Рис.1. Дерево всех партий, возможных при Ваниной стратегии. Знаком >> обозначены позиции, в которых партия заканчивается.

Указания по оцениванию	Баллы
В задаче от ученика требуется выполнить 3 задания. Их трудность возрастает. Количество баллов в целом соответствует количеству выполненных заданий (подробнее см. ниже). Ошибка в решении, не искажающая основного замысла, например, арифметическая ошибка при вычислении количества камней в заключительной позиции, при оценке решения не учитывается. Первое задание считается выполненным полностью, если выполнены полностью оба пункта а) и б). Пункт а) считается выполненным полностью, если правильно указаны все позиции, в которых Петя выигрывает первым ходом и указано, каким должен быть первый ход. Пункт б) считается выполненным, если правильно указана позиция, в которой Ваня выигрывает первым ходом и описана стратегия Вани, т.е. показано, как Ваня может получить кучу, в которой содержится нужное количество камней при любом ходе Пети. Второе задание выполнено, если правильно указаны обе позиции, выигрышная для Пети и описана соответствующая стратегия Пети – так, как это написано в примере решения или другим способом, например, с помощью дерева всех возможных партий. Третье задание выполнено, если правильно указана позиция, выигрышная для Вани и построено дерево всех партий, возможных при Ваниной стратегии. Должно быть явно сказано, что в этом дереве в каждой позиции, где должен ходить Петя, разобраны все возможные ходы, а для позиций, где должен ходить Ваня – только ход, соответствующий стратегии, которую выбрал Ваня. Во всех случаях стратегии могут быть описаны так, как это сделано в примере решения или другим способом.	
Выполнены второе и третье задания. Первое задание выполнено полностью или частично. Здесь и далее допускаются арифметические ошибки, которые не искажают сути решения и не приводят к неправильному ответу (см. выше).	3
Не выполнены условия, позволяющие поставить 3 балла, и выполнено одно из следующих условий. 1. Задание 3 выполнено полностью. 2. Первое и второе задания выполнены полностью.	2
Не выполнены условия, позволяющие поставить 3 или 2 балла, и выполнено одно из следующих условий. 1. Первое задание выполнено полностью. 2. Во втором задании правильно указано одно из двух возможных значений S, и для этого значения указана и обоснована выигрышная стратегия Пети.	1
Не выполнено ни одно из условий, позволяющих поставить 3, 2 или 1 балл	0
Максимальный балл	*3*

Тренинг по оцениванию задания 27.

Задание 27 проверяет умение создавать собственные программы (30–50 строк) для решения задач средней сложности.

Разберите варианты решений, представленных учащимися, и проведите процедуру оценивания в соответствии с критериями.

27 В физической лаборатории проводится долговременный эксперимент по изучению гравитационного поля Земли. По каналу связи каждую минуту в лабораторию передаётся положительное целое число – текущее показание прибора «Сигма 2015». Количество передаваемых чисел в серии известно и не превышает 10 000. Все числа не превышают 1000. Временем, в течение которого происходит передача, можно пренебречь.

Необходимо вычислить «бета-значение» серии показаний прибора – минимальное чётное произведение двух показаний, между моментами передачи которых прошло не менее 7 минут. Если получить такое произведение не удаётся, ответ считается равным –1.

Входные данные представлены следующим образом. В первой строке задаётся число N – общее количество показаний прибора. Гарантируется, что N > 7. В каждой из следующих N строк задаётся одно положительное целое число – очередное показание прибора.

Требуется написать эффективную по времени и по памяти программу для решения описанной задачи. Программа считается эффективной по времени, если при увеличении количества исходных чисел N в k раз время работы программы увеличивается не более чем в k раз.
Программа считается эффективной по памяти, если память, необходимая для хранения всех переменных программы, не превышает 1 Кбайт и не увеличивается с ростом N.
Максимальная оценка за правильную (не содержащую синтаксических ошибок и дающую правильный ответ при любых допустимых входных данных) программу, эффективную по времени и по памяти, – 4 балла. Максимальная оценка за правильную программу, эффективную только по времени – 3 балла.
Максимальная оценка за правильную программу, не удовлетворяющую требованиям эффективности, – 2 балла.
Вы можете сдать одну программу или две программы решения задачи (например, одна из программ может быть менее эффективна). Если Вы сдадите две программы, то каждая из них будет оцениваться независимо от другой, итоговой станет бóльшая из двух оценок.
Перед текстом программы обязательно кратко опишите алгоритм решения. Укажите использованный язык программирования и его версию.

Пример входных данных:
12
12
45
5
3
14
17
23
21
20
19
18
17
Программа должна вывести одно число – описанное в условии произведение, либо –1, если получить такое произведение не удаётся.
Пример выходных данных для приведённого выше примера входных данных:
54

Содержание верного ответа и указания по оцениванию (допускаются иные формулировки ответа, не искажающие его смысла)
Эффективное по времени и памяти решение. Чтобы произведение было чётным, хотя бы один сомножитель должен быть чётным, поэтому при поиске подходящих произведений чётные показания прибора можно рассматривать в паре с любыми другими, а нечётные – только с чётными. Для каждого показания с номером *k*, начиная с *k* = 8, рассмотрим все допустимые по условиям задачи пары, в которых данное показание получено вторым. Минимальное произведение из всех этих пар будет получено, если первым в паре будет взято

минимальное подходящее показание среди всех, полученных от начала приёма и до показания с номером *k* – 7. Если очередное показание чётное, минимальное среди предыдущих может быть любым, если нечётное – только чётным.

Для получения эффективного по времени решения нужно по мере ввода данных помнить абсолютное минимальное и минимальное чётное показание на каждый момент времени, каждое вновь полученное показание умножать на соответствующий ему минимум, имевшийся на 7 элементов ранее, и выбрать минимальное из всех таких произведений.
Поскольку каждое текущее минимальное показание используется после ввода ещё 7 элементов и после этого становится ненужным, достаточно хранить только 7 последних минимумов. Для этого можно использовать массив из 7 элементов и циклически заполнять его по мере ввода данных. Размер этого массива не зависит от общего количества введённых показаний, поэтому такое решение будет эффективным не только по времени, но и по памяти. Чтобы хранить абсолютный и чётный минимумы, нужно использовать два таких массива.
Ниже приводится пример такой программы, написанной на алгоритмическом языке.

Программа 1. Пример правильной программы на алгоритмическом языке.
Программа эффективна по времени и по памяти

```
алг
нач
 цел s = 7 | требуемое расстояние между показаниями
 цел amax = 1001 | больше максимально возможного показания
 цел N
 ввод N
 цел a | очередное показание прибора
 целтаб мини[0:s-1] | текущие минимумы последних s элементов
 целтаб миничет[0:s-1] | чётные минимумы последних s элементов
 цел i
 | вводим первые s показаний, фиксируем минимумы
 цел ма; ма := amax | минимальное показание
 цел мчет; мчет := amax | минимальное чётное показание
 нц для i от 1 до s
 ввод a
 ма := imin(ма, a)
 если mod(a,2) = 0 то мчет := imin(мчет,a) все
 мини[mod(i, s)] := ма
 миничет[mod(i, s)] := мчет
 кц
 цел мп = amax*amax | минимальное значение произведения
 цел п
 нц для i от s+1 до N
 ввод a
 если mod(a,2)=0
 то п := a * мини[mod(i, s)]
 иначе если мчет < amax
 то п := a * миничет[mod(i, s)]
 иначе п := amax*amax;
 все
 все
 мп := imin(мп, п)
 ма := imin(ма, a)
 если mod(a,2) = 0 то мчет := imin(мчет,a) все
 мини[mod(i, s)] := ма
 миничет[mod(i, s)] := мчет
```

```
 кц
 если мп = amax*amax то мп:=-1 все
 вывод мп
кон
```

Возможны и другие реализации. Например, вместо циклического заполнения массива можно каждый раз сдвигать его элементы. В приведённом ниже примере хранятся и сдвигаются не минимумы, а исходные значения. Это требует чуть меньше памяти (достаточно одного массива вместо двух), но по времени решение со сдвигами менее эффективно, чем с циклическим заполнением. Однако время работы остаётся пропорциональным *N*, поэтому максимальная оценка за такое решение тоже составляет 4 балла.

Программа 2. Пример правильной программы на языке Паскаль.
Программа использует сдвиги, но эффективна по времени и по памяти

```
const s = 7; {требуемое расстояние между показаниями}
 amax = 1001; {больше максимально возможного показания}
var
 N: integer;
 a: array[1..s] of integer; {хранение s показаний прибора}
 a_: integer; {ввод очередного показания}
 ma: integer; {минимальное число без s последних}
 me: integer; {минимальное чётное число без s последних}
 mp: integer; {минимальное значение произведения}
 p: integer;
 i, j: integer;
begin
 readln(N);
 {Ввод первых s чисел}
 for i:=1 to s do readln(a[i]);
 {Ввод остальных значений, поиск минимального произведения}
 ma := amax; me := amax;
 mp :=amax*amax;
 for i := s + 1 to N do begin
 readln(a_);
 if a[1] < ma then ma := a[1];
 if (a[1] mod 2 = 0) and (a[1] < me) then me := a[1];
 if a_ mod 2 = 0 then p := a_ * ma
 else if me < amax then p := a_ * me
 else p := amax* amax;
 if (p < mp) then mp := p;
 {сдвигаем элементы вспомогательного массива влево}
 for j := 1 to s - 1 do
 a[j] := a[j + 1];
 a[s] := a_
 end;
 if mp = amax*amax then mp:=-1;
 writeln(mp)
end.
```

Если вместо небольшого массива фиксированного размера (циклического или со сдвигами) хранятся все исходные данные (или все текущие минимумы), программа сохраняет эффективность по времени, но становится неэффективной по памяти, так как требуемая память растёт пропорционально *N*. Ниже приводится пример такой программы на языке Паскаль. Подобные (и аналогичные по сути) программы оцениваются не выше 3 баллов.

Программа 3. Пример правильной программы на языке Паскаль.
Программа эффективна по времени, но неэффективна по памяти

```
const s = 7; {требуемое расстояние между показаниями}
 amax = 1001; {больше максимально возможного показания}
var
 N, p, i: integer;
 a: array[1..10000] of integer; {все показания прибора}
 ma: integer; {минимальное число без s последних}
 me: integer; {минимальное чётное число без s последних}
 mp: integer; {минимальное значение произведения}
begin
 readln(N);
 {Ввод всех показаний прибора}
 for i:=1 to N do readln(a[i]);
 ma := amax;
 me := amax;
 mp := amax*amax;
 for i := s + 1 to N do
 begin
 if a[i-s] < ma then ma := a[i-s];
 if (a[i-s] mod 2 = 0) and (a[i-s] < me) then
 me := a[i-s];
 if a[i] mod 2 = 0 then p := a[i] * ma
 else if me < amax then p := a[i] * me
 else p := amax * amax;
 if (p < mp) then mp := p
 end;
 if mp = amax*amax then mp := -1;
 writeln(mp)
end.
```

Возможно также переборное решение, в котором находятся произведения всех возможных пар и из них выбирается минимальное. Ниже (см. программу 4) приведён пример подобного решения. Это (и аналогичные ему) решение неэффективно ни по времени, ни по памяти. Оценка за такое решение – 2 балла.

Программа 4. Пример правильной программы на языке Паскаль.
Программа неэффективна ни по времени, ни по памяти

```
const s = 7; {требуемое расстояние между показаниями}
var
 N: integer;
 a: array[1..10000] of integer; {все показания прибора}
 mp: integer; {минимальное значение произведения}
 i, j: integer;
begin
 readln(N);
 {Ввод значений прибора}
 for i:=1 to N do
 readln(a[i]);
 mp := 1000 * 1000 + 1;
 for i := 1 to N-s do begin
 for j := i+s to N do begin
 if (a[i]*a[j] mod 2 = 0) and (a[i]*a[j] < mp)
 then mp := a[i]*a[j]
```

```
 end;
 end;
 if mp = 1000 * 1000 + 1 then mp := -1;
 writeln(mp)
end.
```

Указания по оцениванию	Баллы
Критерии оценивания неэффективной программы	
Программа решает поставленную задачу для любых соответствующих условию входных данных. Например, допускается переборное решение, аналогичное приведённой выше программе 4. Допускается до семи синтаксических и приравненных к ним ошибок (см. критерии оценивания на 4 балла). Допускается до двух содержательных ошибок, описанных в критериях оценивания на 3 балла	2
Не выполнены условия, позволяющие поставить 2 балла. Из программы видно, что экзаменуемый в целом правильно представляет путь решения задачи независимо от эффективности.	1
Не выполнены критерии, позволяющие поставить 1 или 2 балла	0
Максимальный балл для задания А	*2*

Критерии оценивания эффективной программы	
Программа правильно работает для любых соответствующих условию входных данных и при этом эффективна как по времени, так и по памяти, т.е. не используются массивы и другие структуры данных, размер которых зависит от количества входных элементов, а время работы пропорционально этому количеству. Возможно использование массивов и динамических структур данных (например, контейнеры STL в программе на языке C++) при условии, что в них в каждый момент времени хранится фиксированное количество элементов, требующих для хранения меньше 1кб (минимально необходимое количество – семь; допускается решение с запасом). Программа может содержать не более трёх синтаксических ошибок следующих видов: – пропущен или неверно указан знак пунктуации (запятая, точка с запятой, скобки и т.д.); – неверно написано или пропущено служебное слово языка программирования; – не описана или неверно описана переменная; – применяется операция, недопустимая для соответствующего типа данных. К синтаксическим ошибкам приравнивается использование неверного типа данных (например, использование целого типа вместо вещественного для представления данных при вводе и обработке). Если одна и та же ошибка встречается несколько раз, она считается за одну ошибку	4
Не выполнены условия, позволяющие поставить 4 балла. Программа правильно работает для любых соответствующих условию входных данных, время работы пропорционально количеству входных элементов. Размер используемой памяти не имеет значения и может зависеть от объёма входных данных. В частности, допускается использование одного или нескольких массивов размера *N* (как в приведённой выше программе 3). Программа может содержать не более пяти синтаксических и приравненных к ним ошибок, описанных в критериях на 4 балла.	3

Кроме того, допускается наличие не более одной содержательной ошибки из числа следующих: неверная инициализация при поиске минимального значения; неверная обработка начальных элементов данных, которая может, например, привести к получению ошибочного ответа при 7 < *N* < 14; неточное определение границ массива, выход за границу массива (например, описан массив с границами от 1 до 7, а реально используется от 0 до 6 или наоборот); вычисленный индекс элемента массива на 1 отличается от верного; используется операция ”<” вместо ”<=”, ”or” вместо ”and” и т.п.; не учитывается, что заданные показания могут начинаться с одного или нескольких чётных чисел; не учитывается, что для данного набора показаний может не быть ни одного удовлетворяющего условиям произведения; использована одна переменная (или константа) вместо другой; используется один знак операции вместо другого; используется одно зарезервированное слово языка программирования вместо другого.	

Не выполнены условия, позволяющие поставить 3 или 4 балла. Программа работает в целом верно, эффективно или нет. Например, допускается переборное решение, аналогичное приведённой выше программе 4. Допускается до семи синтаксических и приравненных к ним ошибок (см. критерии на 4 балла). Допускается до двух содержательных ошибок, описанных в критериях на 3 балла	2
Не выполнены условия, позволяющие поставить 2, 3 или 4 балла. Из программы видно,что экзаменуемый в целом правильно представляет путь решения задачи независимо от эффективности.	1
Не выполнены критерии, позволяющие поставить 1, 2, 3 или 4 балла	0
Максимальный балл для задания Б	*4*
Итоговый максимальный балл	*4*

Литература

Материалы ФИПИ: http://fipi.ru/ege-i-gve-11/dlya-predmetnyh-komissiy-subektov-rf

IV. Кадровое обеспечение дополнительной образовательной программы

№ п/п	Дисциплины (модули)	Характеристика педагогических работников							
		фамилия, имя, отчество, должность по штатному расписанию	Какое образовательное учреждение окончил, специальность (направление подготовки)	Ученая степень, ученое (почетное) звание, квалификационная категория	стаж педагогический (научно-педагогической) работы			основное место работы, должность	условия привлечения к педагогической деятельности
					всего	в т.ч. педагогической работы			
						всего	в т.ч. по указанной дисциплине		
1	2	3	4	5	6	7	8	9	10
1	Структура КИМов единого государственного	Каплиева Наталья Алексеевна,	Воронежский государственный	Канд. физ.-мат.	26	18	18	Воронежский государ	Штатный работн

	экзамена по информатике и ИКТ	доцент	университет, прикладная математика	наук				ственный университет, доцент	ик
2	Характеристика заданий первой части КИМов единого государственного экзамена по информатике и ИКТ	Каплиева Наталья Алексеевна, доцент	Воронежский государственный университет, прикладная математика	Канд. физ.-мат. наук	26	18	18	Воронежский государственный университет, доцент	Штатный работник
3	Характеристика заданий второй части КИМов единого государственного экзамена по информатике и ИКТ	Каплиева Наталья Алексеевна, доцент	Воронежский государственный университет, прикладная математика	Канд. физ.-мат. наук	26	18	18	Воронежский государственный университет, доцент	Штатный работник
4	Критерии оценки заданий второй части КИМов единого государственного экзамена по информатике и ИКТ	Каплиева Наталья Алексеевна, доцент	Воронежский государственный университет, прикладная математика	Канд. физ.-мат. наук	26	18	18	Воронежский государственный университет, доцент	Штатный работник

V. Составители программы

Каплиева Наталья Алексеевна, канд. физ.-мат. наук, доцент