

## ПРОЕКТ

Единый государственный экзамен по ИНФОРМАТИКЕ и ИКТ

**Демонстрационный вариант**  
контрольных измерительных материалов единого  
государственного экзамена 2020 года  
по информатике и ИКТ

подготовлен Федеральным государственным бюджетным  
научным учреждением

«ФЕДЕРАЛЬНЫЙ ИНСТИТУТ ПЕДАГОГИЧЕСКИХ ИЗМЕРЕНИЙ»

### Единый государственный экзамен по ИНФОРМАТИКЕ и ИКТ

#### **Пояснения к демонстрационному варианту контрольных измерительных материалов единого государственного экзамена 2020 года по ИНФОРМАТИКЕ и ИКТ**

При ознакомлении с демонстрационным вариантом контрольных измерительных материалов ЕГЭ 2020 г. следует иметь в виду, что задания, включённые в него, не отражают всех вопросов содержания, которые будут проверяться с помощью вариантов КИМ в 2020 г. Полный перечень вопросов, которые могут контролироваться на едином государственном экзамене 2020 г., приведён в кодификаторе элементов содержания и требований к уровню подготовки выпускников образовательных организаций для проведения единого государственного экзамена 2020 г. по информатике и ИКТ.

Назначение демонстрационного варианта заключается в том, чтобы дать возможность любому участнику ЕГЭ и широкой общественности составить представление о структуре будущих КИМ, количестве заданий, об их форме и уровне сложности. Приведённые критерии оценки выполнения заданий с развёрнутым ответом, включённые в этот вариант, дают представление о требованиях к полноте и правильности записи развёрнутого ответа.

Эти сведения позволят выпускникам выработать стратегию подготовки к ЕГЭ в 2020 г.



**Часть 1**

Ответами к заданиям 1–23 являются число, последовательность букв или цифр, которые следует записать в БЛАНК ОТВЕТОВ № 1 справа от номера соответствующего задания, начиная с первой клеточки, без пробелов, запятых и других дополнительных символов. Каждый символ пишете в отдельной клеточке в соответствии с приведёнными в бланке образцами.

**1** Сколько существует натуральных чисел  $x$ , для которых выполняется неравенство  $10011011_2 < x < 10011111_2$ ?  
В ответе укажите только количество чисел, сами числа писать не нужно.

Ответ: \_\_\_\_\_.

**2** Миша заполнял таблицу истинности функции  $(x \wedge \neg y) \vee (x \equiv z) \vee \neg w$ , но успел заполнить лишь фрагмент из трёх **различных** её строк, даже не указав, какому столбцу таблицы соответствует каждая из переменных  $w, x, y, z$ .

				$(x \wedge \neg y) \vee (x \equiv z) \vee \neg w$
0	1	1	0	0
0				0
	1	0	1	0

Определите, какому столбцу таблицы соответствует каждая из переменных  $w, x, y, z$ .

В ответе напишите буквы  $w, x, y, z$  в том порядке, в котором идут соответствующие им столбцы (сначала буква, соответствующая первому столбцу; затем буква, соответствующая второму столбцу, и т.д.). Буквы в ответе пишите подряд, никаких разделителей между буквами ставить не нужно.

Пример. Функция задана выражением  $\neg x \vee y$ , зависящим от двух переменных, а фрагмент таблицы имеет следующий вид.

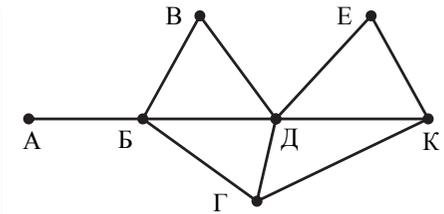
		$\neg x \vee y$
0	1	0

В этом случае первому столбцу соответствует переменная  $y$ , а второму столбцу – переменная  $x$ . В ответе следует написать  $yx$ .

Ответ: \_\_\_\_\_.

**3** На рисунке схема дорог Н-ского района изображена в виде графа, в таблице содержатся сведения о протяжённости каждой из этих дорог (в километрах).

		Номер пункта						
		1	2	3	4	5	6	7
Номер пункта	1				9			7
	2				5		11	
	3						12	
	4	9	5			4	13	15
	5				4		10	8
	6		11	12	13	10		
	7	7			15	8		



Так как таблицу и схему рисовали независимо друг от друга, то нумерация населённых пунктов в таблице никак не связана с буквенными обозначениями на графе. Определите, какова протяжённость дороги из пункта Д в пункт Е. В ответе запишите целое число – так, как оно указано в таблице.

Ответ: \_\_\_\_\_.

4 Ниже представлены два фрагмента таблиц из базы данных о жителях микрорайона. Каждая строка таблицы 2 содержит информацию о ребёнке и об одном из его родителей. Информация представлена значением поля ID в соответствующей строке таблицы 1. Определите на основании приведённых данных, у скольких детей на момент их рождения отцам было больше 25 полных лет. При вычислении ответа учитывайте только информацию из приведённых фрагментов таблиц.

Таблица 1			
ID	Фамилия_И.О.	Пол	Год_рождения
14	Краснова Н.А.	Ж	1942
24	Сканави И.П.	М	1943
25	Сканави П.И.	М	1973
26	Сканави П.П.	М	1996
34	Кущенко А.И.	Ж	1964
35	Кущенко В.С.	Ж	1987
36	Кущенко С.С.	М	1964
44	Лебедь А.С.	Ж	1941
45	Лебедь В.А.	М	1953
46	Гросс О.С.	Ж	1992
47	Гросс П.О.	М	2009
54	Клычко А.П.	Ж	1993
64	Крот П.А.	Ж	1964
...	...	...	...

Таблица 2	
ID_Родителя	ID_Ребёнка
24	25
44	25
25	26
64	26
24	34
44	34
34	35
36	35
14	36
34	46
36	46
25	54
64	54
...	...

Ответ: \_\_\_\_\_.

5 Для кодирования некоторой последовательности, состоящей из букв К, Л, М, Н, П, Р, решили использовать неравномерный двоичный код, удовлетворяющий условию Фано. Для букв К, Л, М, Н использовали соответственно кодовые слова 000, 001, 010, 11. Для двух оставшихся букв – П и Р – длины кодовых слов неизвестны. Укажите кратчайшее возможное кодовое слово для буквы П, при котором код будет удовлетворять условию Фано. Если таких кодов несколько, укажите код с **наименьшим** числовым значением.  
*Примечание.* Условие Фано означает, что никакое кодовое слово не является началом другого кодового слова. Это обеспечивает возможность однозначной расшифровки закодированных сообщений.

Ответ: \_\_\_\_\_.

6 На вход алгоритма подаётся натуральное число  $N$ . Алгоритм строит по нему новое число  $R$  следующим образом.

1. Строится двоичная запись числа  $N$ .
2. К этой записи дописываются справа ещё два разряда по следующему правилу:
  - а) складываются все цифры двоичной записи числа  $N$ , и остаток от деления суммы на 2 дописывается в конец числа (справа). Например, запись 11100 преобразуется в запись 111001;
  - б) над этой записью производятся те же действия – справа дописывается остаток от деления суммы её цифр на 2.

Полученная таким образом запись (в ней на два разряда больше, чем в записи исходного числа  $N$ ) является двоичной записью искомого числа  $R$ . Укажите минимальное число  $R$ , которое превышает число 97 и может являться результатом работы данного алгоритма. В ответе это число запишите в десятичной системе счисления.

Ответ: \_\_\_\_\_.

7 Дан фрагмент электронной таблицы. Из ячейки В3 в ячейку С2 была скопирована формула. При копировании адреса ячеек в формуле автоматически изменились. Каким стало числовое значение формулы в ячейке С2?

	A	B	C	D	E
1	1	10	100	1000	10000
2	2	20		2000	20000
3	3	=A\$2+D\$3	300	3000	30000
4	4	40	400	4000	40000

*Примечание.* Знак \$ обозначает абсолютную адресацию.

Ответ: \_\_\_\_\_.

**8** Запишите число, которое будет напечатано в результате выполнения следующей программы. Для Вашего удобства программа представлена на пяти языках программирования.

<b>Бейсик</b>	<b>Python</b>
<pre>DIM S, N AS INTEGER S = 0 N = 1 WHILE S &lt; 51   S = S + 11   N = N * 2 WEND PRINT N</pre>	<pre>s = 0 n = 1 while s &lt; 51:   s = s + 11   n = n * 2 print(n)</pre>
<b>Алгоритмический язык</b>	<b>Паскаль</b>
<pre>алг нач   цел n, s   s := 0   n := 1   нц пока s &lt; 51     s := s + 11     n := n * 2   кц   вывод n кон</pre>	<pre>var s, n: integer; begin   s := 0;   n := 1;   while s &lt; 51 do     begin       s := s + 11;       n := n * 2;     end;   writeln(n); end.</pre>
<b>C++</b>	
<pre>#include &lt;iostream&gt; using namespace std;  int main() {   int s = 0, n = 1;   while (s &lt; 51) { s = s + 11; n = n * 2; }   cout &lt;&lt; n &lt;&lt; endl;   return 0; }</pre>	

Ответ: \_\_\_\_\_.

**9** Для хранения произвольного растрового изображения размером 128×320 пикселей отведено 40 Кбайт памяти без учёта размера заголовка файла. Для кодирования цвета каждого пикселя используется одинаковое количество бит, коды пикселей записываются в файл один за другим без промежутков. Какое максимальное количество цветов можно использовать в изображении?

Ответ: \_\_\_\_\_.

**10** Все 4-буквенные слова, в составе которых могут быть буквы Н, О, Т, К, И, записаны в алфавитном порядке и пронумерованы, начиная с 1.

Ниже приведено начало списка.

1. ИИИИ
2. ИИИК
3. ИИИН
4. ИИИО
5. ИИИТ
6. ИИКИ

...

Под каким номером в списке идёт первое слово, которое начинается с буквы О?

Ответ: \_\_\_\_\_.

11

Ниже на пяти языках программирования записан рекурсивный алгоритм F.

<b>Бейсик</b>	<b>Python</b>
<pre>SUB F(n)   PRINT n,   IF n &gt;= 3 THEN     F(n \ 2)     F(n - 1)   END IF END SUB</pre>	<pre>def F(n):   print(n, end='')   if n &gt;= 3:     F(n // 2)     F(n - 1)</pre>
<b>Алгоритмический язык</b>	<b>Паскаль</b>
<pre>алг F(цел n) нач   вывод n   если n &gt;= 3 то     F(div(n, 2))     F(n - 1)   все кон</pre>	<pre>procedure F(n: integer); begin   write(n);   if n &gt;= 3 then     begin       F(n div 2);       F(n - 1)     end end;</pre>
<b>C++</b>	
<pre>void F(int n) {   std::cout &lt;&lt; n;   if (n &gt;= 3) {     F(n / 2);     F(n - 1);   } }</pre>	

Запишите подряд без пробелов и разделителей все числа, которые будут выведены на экран при выполнении вызова F(5). Числа должны быть записаны в том же порядке, в котором они выводятся на экран.

Ответ: \_\_\_\_\_.

12

В терминологии сетей TCP/IP маской сети называется двоичное число, определяющее, какая часть IP-адреса узла сети относится к адресу сети, а какая – к адресу самого узла в этой сети. Обычно маска записывается по тем же правилам, что и IP-адрес, – в виде четырёх байтов, причём каждый байт записывается в виде десятичного числа. При этом в маске сначала (в старших разрядах) стоят единицы, а затем с некоторого разряда – нули. Адрес сети получается в результате применения поразрядной конъюнкции к заданному IP-адресу узла и маске.

Например, если IP-адрес узла равен 231.32.255.131, а маска равна 255.255.240.0, то адрес сети равен 231.32.240.0.

Для узла с IP-адресом 111.81.27.224 адрес сети равен 111.81.27.192. Чему равен последний (самый правый) байт маски? Ответ запишите в виде десятичного числа.

Ответ: \_\_\_\_\_.

13

При регистрации в компьютерной системе каждому пользователю выдаётся пароль, состоящий из 15 символов и содержащий только символы из 8-символьного набора: A, B, C, D, E, F, G, H. В базе данных для хранения сведений о каждом пользователе отведено одинаковое и минимально возможное целое число байт. При этом используют посимвольное кодирование паролей, все символы кодируют одинаковым и минимально возможным количеством бит. Кроме собственно пароля, для каждого пользователя в системе хранятся дополнительные сведения, для чего отведено 24 байт на одного пользователя.

Определите объём памяти (в байтах), необходимый для хранения сведений о 20 пользователях. В ответе запишите только целое число – количество байт.

Ответ: \_\_\_\_\_.

14

Исполнитель Редактор получает на вход строку цифр и преобразовывает её. Редактор может выполнять две команды, в обеих командах  $v$  и  $w$  обозначают цепочки цифр.

А) **заменить** ( $v, w$ ).

Эта команда заменяет в строке первое слева вхождение цепочки  $v$  на цепочку  $w$ . Например, выполнение команды

**заменить** (111, 27)

преобразует строку 05111150 в строку 0527150.

Если в строке нет вхождений цепочки  $v$ , то выполнение команды

**заменить** ( $v, w$ ) не меняет эту строку.

Б) **нашлось** ( $v$ ).

Эта команда проверяет, встречается ли цепочка  $v$  в строке исполнителя Редактор. Если она встречается, то команда возвращает логическое значение «истина», в противном случае возвращает значение «ложь». Строка исполнителя при этом не изменяется.

Цикл

ПОКА *условие*

*последовательность команд*

КОНЕЦ ПОКА

выполняется, пока условие истинно.

В конструкции

ЕСЛИ *условие*

ТО *команда1*

ИНАЧЕ *команда2*

КОНЕЦ ЕСЛИ

выполняется *команда1* (если условие истинно) или *команда2* (если условие ложно).

Какая строка получится в результате применения приведённой ниже программы к строке, состоящей из 70 идущих подряд цифр 8? В ответе запишите полученную строку.

НАЧАЛО

ПОКА **нашлось** (2222) ИЛИ **нашлось** (8888)

ЕСЛИ **нашлось** (2222)

ТО **заменить** (2222, 88)

ИНАЧЕ **заменить** (8888, 22)

КОНЕЦ ЕСЛИ

КОНЕЦ ПОКА

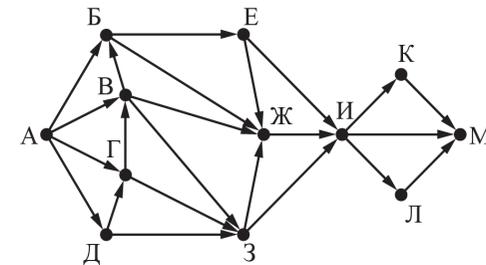
КОНЕЦ

Ответ: \_\_\_\_\_.

15

На рисунке представлена схема дорог, связывающих города А, Б, В, Г, Д, Е, Ж, З, И, К, Л, М. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой.

Сколько существует различных путей из города А в город М, проходящих через город Ж?



Ответ: \_\_\_\_\_.

16

Сколько единиц содержится в двоичной записи значения выражения:  $4^8 + 2^8 - 8$ ?

Ответ: \_\_\_\_\_.

17

В языке запросов поискового сервера для обозначения логической операции «ИЛИ» используется символ «|», а для обозначения логической операции «И» – символ «&».

В таблице приведены запросы и количество найденных по ним страниц некоторого сегмента сети Интернет.

Запрос	Найдено страниц (в сотнях тысяч)
Поле	54
Пшеница	40
Напряжённость	44
Поле & Пшеница	30
Напряжённость & Поле	14
Напряжённость & Пшеница	0

Какое количество страниц (в сотнях тысяч) будет найдено по запросу *Напряжённость | Поле | Пшеница*?

Считается, что все запросы выполнялись практически одновременно, так что набор страниц, содержащих все искомые слова, не изменялся за время выполнения запросов.

Ответ: \_\_\_\_\_.

18

Для какого наименьшего целого неотрицательного числа *A* выражение

$$(x + 2y < A) \vee (y > x) \vee (x > 30)$$

тождественно истинно, т.е. принимает значение 1 при любых целых неотрицательных *x* и *y*?

Ответ: \_\_\_\_\_.

19

В программе используется одномерный целочисленный массив *A* с индексами от 0 до 11. Значения элементов равны 20, 19, 17, 41, 23, 12, 24, 16, 4, 13, 6, 15 соответственно, т.е.  $A[0] = 20$ ,  $A[1] = 19$  и т.д. Определите значение переменной *s* после выполнения следующего фрагмента этой программы (записанного ниже на пяти языках программирования).

Бейсик	Python
<pre>s = 0 n = 0 FOR i = 0 TO 11     IF A(i) &lt;= A(n) THEN         s = s + i         t = A(i)         A(i) = A(n)         A(n) = t     END IF NEXT i</pre>	<pre>s = 0 n = 0 for i in range(12):     if A[i] &lt;= A[n]:         s += i         t = A[i]         A[i] = A[n]         A[n] = t</pre>
Алгоритмический язык	Паскаль
<pre>s := 0; n := 0; нц для i от 0 до 11     если A[i] &lt;= A[n] то         s := s + i;         t := A[i];         A[i] := A[n];         A[n] := t;     все кц</pre>	<pre>s := 0; n := 0; for i := 0 to 11 do     if A[i] &lt;= A[n] then         begin             s := s + i;             t := A[i];             A[i] := A[n];             A[n] := t;         end;</pre>
C++	
<pre>s = 0; n = 0; for (int i = 0; i &lt; 12; i++)     if (A[i] &lt;= A[n]){         s += i;         t = A[i];         A[i] = A[n];         A[n] = t;     }</pre>	

Ответ: \_\_\_\_\_.

20

Ниже на пяти языках программирования записан алгоритм. Получив на вход число  $x$ , этот алгоритм печатает два числа:  $L$  и  $M$ . Укажите **наименьшее** число  $x$ , при вводе которого алгоритм печатает сначала 6, а потом 7.

Бейсик	Python
<pre>DIM X, L, M AS INTEGER INPUT X L = 0 M = 0 WHILE X &gt; 0     M = M + 1     IF X MOD 2 = 0 THEN         L = L + 1     ENDIF     X = X \ 2 WEND PRINT L PRINT M</pre>	<pre>x = int(input()) L = 0 M = 0 while x &gt; 0:     M = M + 1     if x % 2 == 0:         L = L + 1     x = x // 2 print(L) print(M)</pre>
Алгоритмический язык	Паскаль
<pre>алг нач     цел x, L, M     ввод x     L := 0     M := 0     нц пока x &gt; 0         M := M + 1         если mod(x,2) = 0             то                 L := L + 1         все         x := div(x,2)     кц     вывод L, M кон</pre>	<pre>var x, L, M: integer; begin     readln(x);     L := 0;     M := 0;     while x &gt; 0 do         begin             M := M + 1;             if x mod 2 = 0 then                 L := L + 1;             x := x div 2;         end;         writeln(L);         writeln(M);     end.</pre>

C++

```
#include <iostream>
using namespace std;

int main()
{
    int x, L, M;
    cin >> x;
    L = 0;
    M = 0;
    while (x > 0){
        M = M + 1;
        if(x % 2 == 0){
            L = L + 1;
        }
        x = x / 2;
    }
    cout << L << endl << M << endl;
    return 0;
}
```

Ответ: \_\_\_\_\_.

21

Напишите в ответе число, которое будет выведено в результате выполнения следующего алгоритма. Для Вашего удобства алгоритм представлен на пяти языках программирования.

Бейсик
<pre>DIM A, B, T, M, R AS LONG A = -20: B = 20 M = A: R = F(A) FOR T = A TO B     IF F(T) &lt; R THEN         M = T         R = F(T)     END IF NEXT T PRINT M + 27  FUNCTION F(x)     F = 2 * (x * x - 100) * (x * x - 100) + 5 END FUNCTION</pre>

### Python

```
def F(x):
    return 2 * (x * x - 100) * (x * x - 100) + 5
a = -20; b = 20
M = a; R = F(a)
for t in range(a, b + 1):
    if (F(t) < R):
        M = t; R = F(t)
print(M + 27)
```

### Алгоритмический язык

```
алг
нач
    цел a, b, t, M, R
    a := -20; b := 20
    M := a; R := F(a)
    нц для t от a до b
        если F(t) < R то
            M := t; R := F(t)
        все
    кц
    вывод M + 27
кон
```

алг цел F(цел x)

```
нач
    знач := 2 * (x * x - 100) * (x * x - 100) + 5
кон
```

### Паскаль

```
var a, b, t, M, R: longint;
function F(x: longint): longint;
begin
    F := 2 * (x * x - 100) * (x * x - 100) + 5;
end;
begin
    a := -20; b := 20;
    M := a; R := F(a);
    for t := a to b do begin
        if (F(t) < R) then begin
            M := t;
            R := F(t)
        end
    end;
    write(M + 27)
end.
```

### C++

```
#include <iostream>
using namespace std;

long F(long x)
{
    return 2 * (x * x - 100) * (x * x - 100) + 5;
}

int main()
{
    long a, b, t, M, R;
    a = -20; b = 20;
    M = a; R = F(a);
    for (t = a; t <= b; t++) {
        if (F(t) < R) {
            M = t; R = F(t);
        }
    }
    cout << M + 27 << endl;
    return 0;
}
```

Ответ: \_\_\_\_\_.

22

Исполнитель преобразует число на экране.

У исполнителя есть две команды, которым присвоены номера:

**1. Прибавить 1**

**2. Умножить на 2**

Первая команда увеличивает число на экране на 1, вторая умножает его на 2. Программа для исполнителя – это последовательность команд.

Сколько существует программ, для которых при исходном числе 1 результатом является число 20 и при этом траектория вычислений содержит число 10?

Траектория вычислений программы – это последовательность результатов выполнения всех команд программы. Например, для программы **121** при исходном числе 7 траектория будет состоять из чисел 8, 16, 17.

Ответ: \_\_\_\_\_.

**23** Сколько существует различных наборов значений логических переменных  $x_1, x_2, \dots, x_8, y_1, y_2, \dots, y_8$ , которые удовлетворяют всем перечисленным ниже условиям?

$$\begin{aligned}(\neg(x_1 \equiv y_1)) &\equiv (x_2 \equiv y_2) \\ (\neg(x_2 \equiv y_2)) &\equiv (x_3 \equiv y_3) \\ &\dots \\ (\neg(x_7 \equiv y_7)) &\equiv (x_8 \equiv y_8)\end{aligned}$$

В ответе **не нужно** перечислять все различные наборы значений переменных  $x_1, x_2, \dots, x_8, y_1, y_2, \dots, y_8$ , при которых выполнена данная система равенств. В качестве ответа Вам нужно указать количество таких наборов.

Ответ: \_\_\_\_\_.



*Не забудьте перенести все ответы в бланк ответов № 1 в соответствии с инструкцией по выполнению работы. Проверьте, чтобы каждый ответ был записан в строке с номером соответствующего задания.*

**Часть 2**

*Для записи ответов на задания этой части (24–27) используйте БЛАНК ОТВЕТОВ № 2. Запишите сначала номер задания (24, 25 и т. д.), а затем полное решение. Ответы записывайте чётко и разборчиво.*

**24** На обработку поступает натуральное число, не превышающее  $10^9$ . Нужно написать программу, которая выводит на экран минимальную чётную цифру этого числа. Если в числе нет чётных цифр, требуется на экран вывести «NO». Программист написал программу неправильно. Ниже эта программа для Вашего удобства приведена на пяти языках программирования.

Бейсик	Python
<pre> DIM N, DIGIT, MINDIGIT AS LONG INPUT N MINDIGIT = N MOD 10 WHILE N &gt; 0     DIGIT = N MOD 10     IF DIGIT MOD 2 = 0 THEN         IF DIGIT &lt; MINDIGIT THEN             MINDIGIT = DIGIT         END IF     END IF     N = N \ 10 WEND IF MINDIGIT = 0 THEN     PRINT "NO" ELSE     PRINT MINDIGIT END IF                 </pre>	<pre> N = int(input()) minDigit = N % 10 while N &gt; 0:     digit = N % 10     if digit % 2 == 0:         if digit &lt; minDigit:             minDigit = digit     N = N // 10 if minDigit == 0:     print("NO") else:     print(minDigit)                 </pre>

Алгоритмический язык	Паскаль
<pre> алг нач   цел N, digit, minDigit   ввод N   minDigit := mod(N,10)   нц пока N &gt; 0     digit := mod(N,10)     если mod(digit, 2) = 0 то       если digit &lt; minDigit то         minDigit := digit       все     все   N := div(N,10) кц если minDigit = 0 то   вывод "NO" иначе   вывод minDigit все кон </pre>	<pre> var N,digit,minDigit: longint; begin   readln(N);   minDigit := N mod 10;   while N &gt; 0 do   begin     digit := N mod 10;     if digit mod 2 = 0 then       if digit &lt; minDigit then         minDigit := digit;     N := N div 10;   end;   if minDigit = 0 then     writeln('NO')   else     writeln(minDigit)   end. </pre>
<b>C++</b> <pre> #include &lt;iostream&gt; using namespace std;  int main() {   int N, digit, minDigit;   cin &gt;&gt; N;   minDigit = N % 10;   while (N &gt; 0) {     digit = N % 10;     if (digit % 2 == 0)       if (digit &lt; minDigit)         minDigit = digit;     N = N / 10;   }   if (minDigit == 0)     cout &lt;&lt; "NO" &lt;&lt; endl;   else     cout &lt;&lt; minDigit &lt;&lt; endl;   return 0; } </pre>	

Последовательно выполните следующее.

1. Напишите, что выведет эта программа при вводе числа 231.
2. Приведите пример такого трёхзначного числа, при вводе которого приведённая программа, несмотря на ошибки, выдаёт верный ответ.
3. Найдите допущенные программистом ошибки и исправьте их. Исправление ошибки должно затрагивать только строку, в которой находится ошибка. Для каждой ошибки:
  - 1) выпишите строку, в которой сделана ошибка;
  - 2) укажите, как исправить ошибку, т.е. приведите правильный вариант строки.

Известно, что в тексте программы можно исправить ровно две строки так, чтобы она стала работать правильно.

Достаточно указать ошибки и способ их исправления для одного языка программирования.

Обратите внимание на то, что требуется найти ошибки в имеющейся программе, а не написать свою, возможно, использующую другой алгоритм решения.

25

Дан целочисленный массив из 30 элементов. Элементы массива могут принимать натуральные значения от 1 до 10 000 включительно. Опишите на одном из языков программирования алгоритм, который находит минимум среди элементов массива, не делящихся нацело на 6, а затем заменяет каждый элемент, не делящийся нацело на 6, на число, равное найденному минимуму. Гарантируется, что хотя бы один такой элемент в массиве есть. В качестве результата необходимо вывести изменённый массив, каждый элемент выводится с новой строки.

Например, для исходного массива из шести элементов:

14  
6  
11  
18  
9  
24

программа должна вывести следующий массив:

9  
6  
9  
18  
9  
24

Исходные данные объявлены так, как показано ниже на примерах для некоторых языков программирования. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать некоторые из описанных переменных.

Бейсик	Python
<pre>CONST N AS INTEGER = 30 DIM A (1 TO N) AS LONG DIM I AS LONG,       J AS LONG,       K AS LONG  FOR I = 1 TO N   INPUT A(I) NEXT I ...</pre>	<pre># допускается также # использовать две # целочисленные переменные j и k a = [] n = 30 for i in range(0, n):   a.append(int(input())) ...</pre>
Алгоритмический язык	Паскаль
<pre>алг нач   цел N = 30   целтаб a[1:N]   цел i, j, k   нц для i от 1 до N     ввод a[i]   кц   ... кон</pre>	<pre>const   N = 30; var   a: array [1..N] of longint;   i, j, k: longint; begin   for i := 1 to N do     readln(a[i]);   ... end.</pre>
C++	
<pre>#include &lt;iostream&gt; using namespace std; const int N = 30; int main() {   long a[N];   long i, j, k;   for (i = 0; i &lt; N; i++)     cin &gt;&gt; a[i];   ...   return 0; }</pre>	

В качестве ответа Вам 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 Free Pascal 2.6). 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).

26

Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежат две кучи камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в одну из куч (по своему выбору) **один** камень или увеличить количество камней в куче в **три раза**. Например, пусть в одной куче 10 камней, а в другой 7 камней; такую позицию в игре будем обозначать (10, 7). Тогда за один ход можно получить любую из четырёх позиций:

(11, 7), (30, 7), (10, 8), (10, 21). Для того чтобы делать ходы, у каждого игрока есть неограниченное количество камней.

Игра завершается в тот момент, когда суммарное количество камней в кучах становится не менее 68. Победителем считается игрок, сделавший последний ход, т.е. первым получивший такую позицию, при которой в кучах будет 68 или больше камней.

В начальный момент в первой куче было 6 камней, во второй куче –  $S$  камней;  $1 \leq S \leq 61$ .

Будем говорить, 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. В описание выигрышной стратегии **не следует** включать ходы играющего по этой стратегии игрока, не являющиеся для него безусловно выигрышными, т.е. не являющиеся выигрышными независимо от игры противника.

Выполните следующие задания.

#### Задание 1

- Укажите все такие значения числа  $S$ , при которых Петя может выиграть за один ход.
- Известно, что Ваня выиграл своим первым ходом после неудачного первого хода Пети. Укажите минимальное значение  $S$ , когда такая ситуация возможна.

#### Задание 2

Укажите такое значение  $S$ , при котором у Пети есть выигрышная стратегия, причём одновременно выполняются два условия:

- Петя не может выиграть за один ход;
- Петя может выиграть своим вторым ходом независимо от того, как будет ходить Ваня.

Для указанного значения  $S$  опишите выигрышную стратегию Пети.

### Задание 3

Укажите значение  $S$ , при котором одновременно выполняются два условия:

- у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети;
- 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).

В узлах дерева указывайте позиции, на рёбрах рекомендуется указывать ходы. Дерево не должно содержать партии, невозможные при реализации выигрывающим игроком своей выигрышной стратегии. Например, полное дерево игры не является верным ответом на это задание.

27

На вход программы поступает последовательность из  $n$  целых положительных чисел. Рассматриваются все пары элементов последовательности  $a_i$  и  $a_j$ , такие что  $i < j$  и  $a_i > a_j$  (первый элемент пары больше второго;  $i$  и  $j$  – порядковые номера чисел в последовательности входных данных). Среди пар, удовлетворяющих этому условию, необходимо найти и напечатать пару с максимальной суммой элементов, которая делится на  $m = 120$ . Если среди найденных пар максимальную сумму имеют несколько, то можно напечатать любую из них.

#### Описание входных и выходных данных

В первой строке входных данных задаётся количество чисел  $n$  ( $2 \leq n \leq 12\,000$ ).

В каждой из последующих  $n$  строк записано одно целое положительное число, не превышающее 10 000.

В качестве результата программа должна напечатать элементы искомой пары. Если таких пар несколько, можно вывести любую из них. Гарантируется, что хотя бы одна такая пара в последовательности есть.

*Пример входных данных:*

6  
60  
140  
61  
100  
300  
59

*Пример выходных данных для приведённого выше примера входных данных:*

140 100

*Пояснение.* Из шести заданных чисел можно составить три пары, сумма элементов которых делится на  $m=120$ :  $60+300$ ,  $140+100$  и  $61+59$ . Во второй и третьей из этих пар первый элемент больше второго, но во второй паре сумма больше.

Требуется написать эффективную по времени и памяти программу для решения описанной задачи.

Программа считается эффективной по времени, если при одновременном увеличении количества элементов последовательности  $n$  и параметра  $m$  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 4 килобайта 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 памяти, – 4 балла.

Максимальная оценка за правильную программу, возможно, неэффективную по памяти или время выполнения которой существенно зависит от величины  $m$ , – 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 **большая** из двух оценок.

Перед текстом программы обязательно кратко опишите алгоритм решения. Укажите использованный язык программирования и его версию.



**Проверьте, чтобы каждый ответ был записан рядом с номером соответствующего задания.**

Система оценивания экзаменационной работы по информатике и ИКТ

Часть 1

За правильный ответ на задания 1–23 ставится 1 балл; за неверный ответ или его отсутствие – 0 баллов.

Номер задания	Правильный ответ
1	3
2	xwzy
3	9
4	2
5	10
6	102
7	30020
8	32
9	256
10	376
11	5242312
12	192
13	600
14	22
15	51
16	6
17	94
18	91
19	16
20	64
21	17
22	28
23	512

Часть 2

Критерии оценивания выполнения заданий с развёрнутым ответом

24

На обработку поступает натуральное число, не превышающее  $10^9$ . Нужно написать программу, которая выводит на экран минимальную чётную цифру этого числа. Если в числе нет чётных цифр, требуется на экран вывести «NO». Программист написал программу неправильно. Ниже эта программа для Вашего удобства приведена на пяти языках программирования.

Бейсик	Python
<pre> DIM N, DIGIT, MINDIGIT AS LONG INPUT N MINDIGIT = N MOD 10 WHILE N &gt; 0     DIGIT = N MOD 10     IF DIGIT MOD 2 = 0 THEN         IF DIGIT &lt; MINDIGIT THEN             MINDIGIT = DIGIT         END IF     END IF     N = N \ 10 WEND IF MINDIGIT = 0 THEN     PRINT "NO" ELSE     PRINT MINDIGIT END IF         </pre>	<pre> N = int(input()) minDigit = N % 10 while N &gt; 0:     digit = N % 10     if digit % 2 == 0:         if digit &lt; minDigit:             minDigit = digit     N = N // 10 if minDigit == 0:     print("NO") else:     print(minDigit)         </pre>
Алгоритмический язык	Паскаль
<pre> алг нач     цел N, digit, minDigit     ввод N     minDigit := mod(N,10)     нц пока N &gt; 0         digit := mod(N,10)         если mod(digit, 2) = 0 то             если digit &lt; minDigit то                 minDigit := digit             все         все         N := div(N,10)     кц     если minDigit = 0 то         вывод "NO"     иначе         вывод minDigit     все кон         </pre>	<pre> var N,digit,minDigit: longint; begin     readln(N);     minDigit := N mod 10;     while N &gt; 0 do         begin             digit := N mod 10;             if digit mod 2 = 0 then                 if digit &lt; minDigit then                     minDigit := digit;                 N := N div 10;             end;         if minDigit = 0 then             writeln('NO')         else             writeln(minDigit)         end.         </pre>

**C++**

```
#include <iostream>
using namespace std;

int main() {
    int N, digit, minDigit;
    cin >> N;
    minDigit = N % 10;
    while (N > 0) {
        digit = N % 10;
        if (digit % 2 == 0)
            if (digit < minDigit)
                minDigit = digit;
        N = N / 10;
    }
    if (minDigit == 0)
        cout << "NO" << endl;
    else
        cout << minDigit << endl;
    return 0;
}
```

Последовательно выполните следующее.

1. Напишите, что выведет эта программа при вводе числа 231.
2. Приведите пример такого трёхзначного числа, при вводе которого приведённая программа, несмотря на ошибки, выдаёт верный ответ.
3. Найдите допущенные программистом ошибки и исправьте их. Исправление ошибки должно затрагивать только строку, в которой находится ошибка. Для каждой ошибки:
  - 1) выпишите строку, в которой сделана ошибка;
  - 2) укажите, как исправить ошибку, т.е. приведите правильный вариант строки.

Известно, что в тексте программы можно исправить ровно две строки так, чтобы она стала работать правильно.

Достаточно указать ошибки и способ их исправления для одного языка программирования.

Обратите внимание на то, что требуется найти ошибки в имеющейся программе, а не написать свою, возможно, использующую другой алгоритм решения.

**Содержание верного ответа и указания по оцениванию**

(допускаются иные формулировки ответа, не искажающие его смысла)

Решение использует запись программы на Паскале. Допускается использование программы на любом из четырёх других языков программирования.

1. Программа выведет число 1.

2. Программа выдаёт правильный ответ, например, для числа 132.

*Замечание для проверяющего. Программа работает неправильно из-за неверной начальной инициализации и неверной проверки отсутствия чётных цифр. Соответственно, программа будет выдавать верный ответ, если вводимое число не содержит 0, содержит хотя бы одну чётную цифру и наименьшая чётная цифра числа не больше младшей (крайней правой) цифры числа (или просто стоит последней).*

3. В программе есть две ошибки.

**Первая ошибка:** неверная инициализация ответа (переменная minDigit).

Строка с ошибкой:

```
minDigit := N mod 10;
```

Верное исправление:

```
minDigit := 10;
```

Вместо 10 может быть использовано любое целое число, большее 8.

**Вторая ошибка:** неверная проверка отсутствия чётных цифр.

Строка с ошибкой:

```
if minDigit = 0 then
```

Верное исправление:

```
if minDigit = 10 then
```

Вместо 10 может быть другое число, большее 8, которое было положено в minDigit при исправлении первой ошибки, или проверка, что minDigit > 8

Указания по оцениванию	Баллы
<p>Обратите внимание! В задаче требовалось выполнить <b>четыре</b> действия:</p> <p>1) указать, что выведет программа при конкретном входном числе;</p> <p>2) указать пример входного числа, при котором программа выдаёт верный ответ;</p> <p>3) исправить первую ошибку;</p> <p>4) исправить вторую ошибку.</p> <p>Для проверки правильности выполнения п. 2) нужно формально выполнить исходную (ошибочную) программу с входными данными, которые указал экзаменуемый, и убедиться в том, что результат, выданный программой, будет таким же, как и для правильной программы.</p> <p>Для действий 3) и 4) ошибка считается исправленной, если выполнены оба следующих условия:</p> <p>а) правильно указана строка с ошибкой;</p> <p>б) указан такой новый вариант строки, что при исправлении другой ошибки получается правильная программа</p>	
Выполнены все четыре необходимых действия, и ни одна верная строка не указана в качестве ошибочной	3
<p>Не выполнены условия, позволяющие поставить 3 балла. Имеет место одна из следующих ситуаций:</p> <p>а) выполнены три из четырёх необходимых действий. Ни одна верная строка не указана в качестве ошибочной;</p> <p>б) выполнены все четыре необходимых действия. Указано в качестве ошибочной не более одной верной строки</p>	2
<p>Не выполнены условия, позволяющие поставить 2 или 3 балла. Выполнены два из четырёх необходимых действий</p>	1
Не выполнены условия, позволяющие поставить 1, 2 или 3 балла	0
<i>Максимальный балл</i>	3

25

Дан целочисленный массив из 30 элементов. Элементы массива могут принимать натуральные значения от 1 до 10 000 включительно. Опишите на одном из языков программирования алгоритм, который находит минимум среди элементов массива, не делящихся нацело на 6, а затем заменяет каждый элемент, не делящийся нацело на 6, на число, равное найденному минимуму. Гарантируется, что хотя бы один такой элемент в массиве есть. В качестве результата необходимо вывести изменённый массив, каждый элемент выводится с новой строки.

Например, для исходного массива из шести элементов:

14  
6  
11  
18  
9  
24

программа должна вывести следующий массив:

9  
6  
9  
18  
9  
24

Исходные данные объявлены так, как показано ниже на примерах для некоторых языков программирования. Запрещается использовать переменные, не описанные ниже, но разрешается не использовать некоторые из описанных переменных.

Бейсик	Python
<pre>CONST N AS INTEGER = 30 DIM A (1 TO N) AS LONG DIM I AS LONG,       J AS LONG,       K AS LONG  FOR I = 1 TO N   INPUT A(I) NEXT I ...</pre>	<pre># допускается также # использовать две # целочисленные переменные j и k a = [] n = 30 for i in range(0, n):   a.append(int(input())) ...</pre>
END	

Алгоритмический язык	Паскаль
<pre> алг нач     цел N = 30     целтаб a[1:N]     цел i, j, k     нц для i от 1 до N         ввод a[i]     кц     ... кон         </pre>	<pre> const     N = 30; var     a: array [1..N] of longint;     i, j, k: longint; begin     for i := 1 to N do         readln(a[i]);     ... end.         </pre>
<b>C++</b>	
<pre> #include &lt;iostream&gt; using namespace std; const int N = 30; int main() {     long a[N];     long i, j, k;     for (i = 0; i &lt; N; i++)         cin &gt;&gt; a[i];     ...     return 0; }         </pre>	

В качестве ответа Вам 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 Free Pascal 2.6). 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)
<b>На языке Паскаль</b>
<pre> k := 10000; for i := 1 to N do     if (a[i] mod 6 &lt;&gt; 0) and (a[i] &lt; k) then         k := a[i]; for i := 1 to N do begin     if (a[i] mod 6 &lt;&gt; 0) then         a[i] := k;     writeln(a[i]); end;         </pre>

На Алгоритмическом языке
<pre> к := 10000 нц для i от 1 до N     если mod(a[i], 6) &lt;&gt; 0 и a[i] &lt; к     то         к := a[i]     все кц нц для i от 1 до N     если mod(a[i], 6) &lt;&gt; 0     то         a[i] := к     все вывод a[i], нс кц         </pre>
<b>На языке Бейсик</b>
<pre> K = 10000 FOR I = 1 TO N     IF A(I) MOD 6 &lt;&gt; 0 AND A(I) &lt; K THEN         K = A(I)     END IF NEXT I FOR I = 1 TO N     IF A(I) MOD 6 &lt;&gt; 0 THEN         A(I) = K     END IF     PRINT A(I) NEXT I         </pre>
<b>На языке C++</b>
<pre> k = 10000; for (i = 0; i &lt; N; i++)     if (a[i] % 6 != 0 &amp;&amp; a[i] &lt; k)         k = a[i]; for (i = 0; i &lt; N; i++) {     if (a[i] % 6 != 0)         a[i] = k;     cout &lt;&lt; a[i] &lt;&lt; endl; }         </pre>
<b>На языке Python</b>
<pre> k = 10000 for i in range(0, n):     if (a[i] % 6 != 0 and a[i] &lt; k):         k = a[i] for i in range(0, n):     if (a[i] % 6 != 0):         a[i] = k print(a[i])         </pre>

Указания по оцениванию	Баллы
<p><i>Общие указания.</i></p> <p>1. В алгоритме, записанном на языке программирования, допускается наличие отдельных синтаксических ошибок, не искажающих замысла автора программы.</p> <p>2. Эффективность алгоритма не имеет значения и не оценивается.</p> <p>3. Допускается запись алгоритма на языке программирования, отличном от языков, приведённых в условии. В этом случае должны использоваться переменные, аналогичные описанным в условии. 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.</p> <p>4. Допускается формат вывода массива, отличный от указанного, например в строчку</p>	
Предложен правильный алгоритм, который изменяет исходный массив и выводит в качестве результата изменённый массив	2
<p>Не выполнены условия, позволяющие поставить 2 балла. При этом предложено в целом верное решение, содержащее не более одной ошибки из числа следующих:</p> <ol style="list-style-type: none"> <li>1) в цикле происходит выход за границу массива;</li> <li>2) не инициализируется или неверно инициализируется минимум;</li> <li>3) неверно осуществляется проверка делимости на 6;</li> <li>4) проверяется делимость на 6 не элемента массива, а его индекса;</li> <li>5) в сравнении с минимумом перепутаны знаки «больше» и «меньше»;</li> <li>6) сравнение с минимумом производится для индекса элемента массива, а не для его значения;</li> <li>7) неверно составлено логическое условие (например, используется <code>or</code> вместо <code>and</code>);</li> <li>8) исходный массив не изменяется;</li> <li>9) изменяются не все требуемые элементы (например, только первый или последний из них);</li> <li>10) отсутствует вывод ответа, или ответ выводится не полностью (например, только один элемент массива ввиду пропущенного цикла вывода элементов или операторных скобок);</li> <li>11) используется переменная, не объявленная в разделе описания переменных;</li> </ol>	1

12) не указано или неверно указано условие завершения цикла; 13) индексная переменная в цикле не меняется (например, в цикле <code>while</code> ) или меняется неверно	
Ошибок, перечисленных в п. 1–13, две или больше, или алгоритм сформулирован неверно (в том числе при отсутствии в явном или неявном виде цикла поиска нужного элемента)	0
<i>Максимальный балл</i>	2

26

Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежат две кучи камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может добавить в одну из куч (по своему выбору) **один** камень или увеличить количество камней в куче в **три раза**. Например, пусть в одной куче 10 камней, а в другой 7 камней; такую позицию в игре будем обозначать (10, 7). Тогда за один ход можно получить любую из четырёх позиций:

(11, 7), (30, 7), (10, 8), (10, 21). Для того чтобы делать ходы, у каждого игрока есть неограниченное количество камней.

Игра завершается в тот момент, когда суммарное количество камней в кучах становится не менее 68. Победителем считается игрок, сделавший последний ход, т.е. первым получивший такую позицию, при которой в кучах будет 68 или больше камней.

В начальный момент в первой куче было 6 камней, во второй куче –  $S$  камней;  $1 \leq S \leq 61$ .

Будем говорить, 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. В описание выигрышной стратегии **не следует** включать ходы играющего по этой стратегии игрока, не являющиеся для него безусловно выигрышными, т.е. не являющиеся выигрышными независимо от игры противника.

Выполните следующие задания.

#### Задание 1

- в) Укажите все такие значения числа  $S$ , при которых Петя может выиграть за один ход.
- г) Известно, что Ваня выиграл своим первым ходом после неудачного первого хода Пети. Укажите минимальное значение  $S$ , когда такая ситуация возможна.

#### Задание 2

Укажите такое значение  $S$ , при котором у Пети есть выигрышная стратегия, причём одновременно выполняются два условия:

- Петя не может выиграть за один ход;
- Петя может выиграть своим вторым ходом независимо от того, как будет ходить Ваня.

Для указанного значения  $S$  опишите выигрышную стратегию Пети.

**Задание 3**

Укажите значение  $S$ , при котором одновременно выполняются два условия:

- у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети;
- 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).

В узлах дерева указывайте позиции, на рёбрах рекомендуется указывать ходы. Дерево не должно содержать партии, невозможные при реализации выигрывающим игроком своей выигрышной стратегии. Например, полное дерево игры не является верным ответом на это задание.

<b>Содержание верного ответа и указания по оцениванию</b> (допускаются иные формулировки ответа, не искажающие его смысла)	
<b>Задание 1</b>	
а) Петя может выиграть при $21 \leq S \leq 61$ .	
б) $S = 7$ .	
<b>Задание 2</b>	
Возможное значение $S$ : 20. В этом случае Петя, очевидно, не может выиграть первым ходом. Однако он может получить позицию (7, 20). После хода Вани может возникнуть одна из четырёх позиций: (8, 20), (21, 20), (7, 21), (7, 60). В каждой из этих позиций Петя может выиграть одним ходом, утроив количество камней во второй куче.	
<i>Замечание для проверяющего.</i> Ещё одно возможное значение $S$ для этого задания – число 13. В этом случае Петя первым ходом должен утроить количество камней в меньшей куче и получить позицию $(6*3, 13) = (18, 13)$ . При такой позиции Ваня не может выиграть первым ходом, а после любого хода Вани Петя может выиграть, утроив количество камней в большей куче. Достаточно указать одно значение $S$ и описать для него выигрышную стратегию.	
<b>Задание 3</b>	
Возможное значение $S$ : 19. После первого хода Пети возможны позиции: (7, 19), (18, 19), (6, 20), (6, 57). В позициях (18, 19) и (6, 57) Ваня может выиграть первым ходом, утроив количество камней во второй куче. Из позиций (7, 19) и (6, 20) Ваня может получить позицию (7, 20). Эта позиция разобрана в п. 2. Игрок, который её получил (теперь это Ваня), выигрывает своим вторым ходом.	
В таблице изображено дерево возможных партий (и только их) при описанной стратегии Вани. Заключительные позиции (в них выигрывает	

Ваня) выделены жирным шрифтом. На рисунке это же дерево изображено в графическом виде (оба способа изображения дерева допустимы).

Исходное положение	Положения после очередных ходов			
	1-й ход Пети (разобраны все ходы, указана полученная позиция)	1-й ход Вани (только ход по стратегии, указана полученная позиция)	2-й ход Пети (разобраны все ходы, указана полученная позиция)	2-й ход Вани (только ход по стратегии, указана полученная позиция)
<b>(6, 19) Всего: 25</b>	(6, 19+1) = (6, 20) Всего: 26	(6+1, 20) = (7, 20) Всего: 27	(7+1, 20) = (8, 20) Всего: 28	(8, 20*3) = (8, 60) <b>Всего: 68</b>
			(7, 20+1) = (7, 21) Всего: 28	(7, 21*3) = (7, 63) <b>Всего: 70</b>
			(7*3, 20) = (21, 20) Всего: 41	(21, 20*3) = (21, 60) <b>Всего: 81</b>
			(7, 20*3) = (7, 60) Всего: 67	(7, 60*3) = (7, 180) <b>Всего: 187</b>
	(6+1, 19) = (7, 19) Всего: 26	(7, 19+1) = (7, 20) Всего: 27	(7+1, 20) = (8, 20) Всего: 28	(8, 20*3) = (8, 60) <b>Всего: 68</b>
			(7, 20+1) = (7, 21) Всего: 28	(7, 21*3) = (7, 63) <b>Всего: 70</b>
			(7*3, 20) = (21, 20) Всего: 41	(21, 20*3) = (21, 60) <b>Всего: 81</b>
			(7, 20*3) = (7, 60) Всего: 67	(7, 60*3) = (7, 180) <b>Всего: 187</b>
	(6*3, 19) = (18, 19) Всего: 37	(18, 19*3) = (18, 57) <b>Всего: 75</b>		
			(6, 19*3) = (6, 57) Всего: 63	(6, 57*3) = (6, 171) <b>Всего: 177</b>

*Примечание для эксперта.* Дерево всех партий может быть также изображено в виде ориентированного графа – так, как показано на рисунке, или другим способом. Важно, чтобы множество полных путей в графе

находилось во взаимно однозначном соответствии со множеством партий, возможных при описанной в решении стратегии.

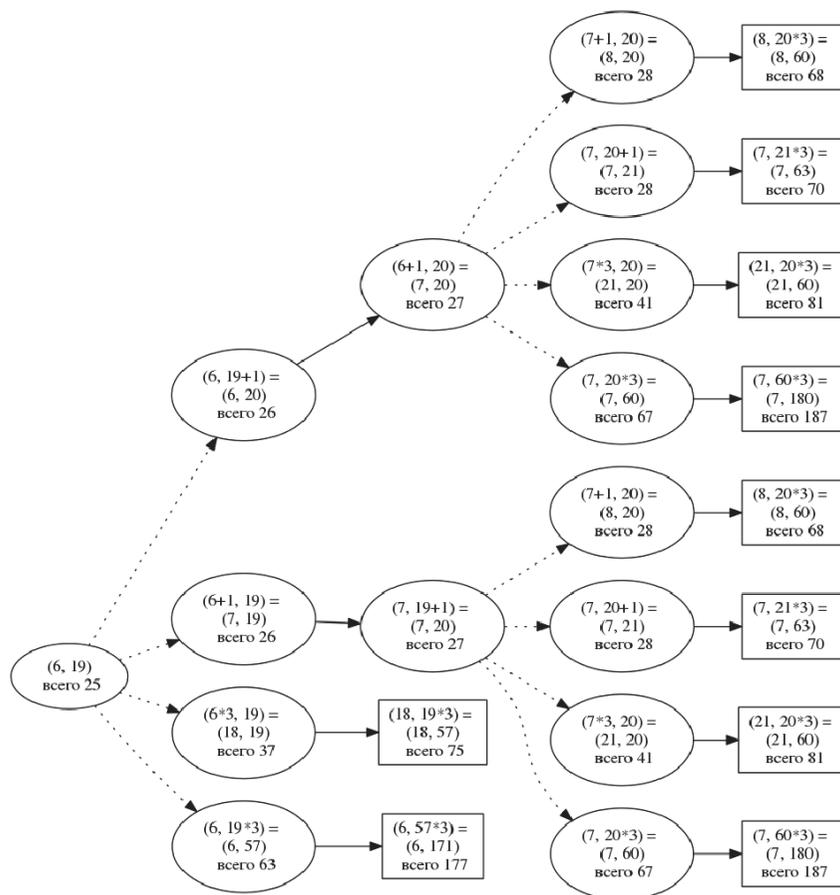


Рис. 1. Дерево всех партий, возможных при Ваниной стратегии. Ходы Пети показаны пунктиром; ходы Вани – сплошными линиями. Прямоугольником обозначены позиции, в которых партия заканчивается.

*Замечание для проверяющего.* Не является ошибкой указание только одного заключительного хода выигрывающего игрока в ситуации, когда у него есть более одного выигрышного хода.

Указания по оцениванию	Баллы
<p>В задаче требуется выполнить три задания. Их трудность возрастает. Количество баллов в целом соответствует количеству выполненных заданий (подробнее см. ниже).</p> <p>Ошибка в решении, не искажающая основного замысла и не приведшая к неверному ответу, например арифметическая ошибка при вычислении количества камней в заключительной позиции, при оценке решения не учитывается.</p> <p>Задание 1 выполнено, если выполнены оба пункта: а) и б), т.е. для п. а) перечислены все значения <math>S</math>, удовлетворяющие условию (и только они), для п. б) указано верное значение <math>S</math> (и только оно).</p> <p>Задание 2 выполнено, если правильно указана позиция, выигрышная для Пети, и описана соответствующая стратегия Пети – так, как это сделано в примере решения, или другим способом, например с помощью дерева всех возможных при выбранной стратегии Пети партий (и только их).</p> <p>Задание 3 выполнено, если правильно указана позиция, выигрышная для Вани, и построено дерево всех возможных при Ваниной стратегии партий (и только их).</p> <p>Во всех случаях стратегии могут быть описаны так, как это сделано в примере решения, или другим способом</p>	
Выполнены задания 1, 2 и 3	3
<p>Не выполнены условия, позволяющие поставить 3 балла, и выполнено одно из следующих условий.</p> <ol style="list-style-type: none"> <li>1. Выполнено задание 3.</li> <li>2. Выполнены задания 1 и 2</li> </ol>	2
<p>Не выполнены условия, позволяющие поставить 2 или 3 балла, и выполнено одно из следующих условий.</p> <ol style="list-style-type: none"> <li>1. Выполнено задание 1.</li> <li>2. Выполнено задание 2</li> </ol>	1
Не выполнено ни одно из условий, позволяющих поставить 1, 2 или 3 балла	0
<i>Максимальный балл</i>	3

27

На вход программы поступает последовательность из  $n$  целых положительных чисел. Рассматриваются все пары элементов последовательности  $a_i$  и  $a_j$ , такие что  $i < j$  и  $a_i > a_j$  (первый элемент пары больше второго;  $i$  и  $j$  – порядковые номера чисел в последовательности входных данных). Среди пар, удовлетворяющих этому условию, необходимо найти и напечатать пару с максимальной суммой элементов, которая делится на  $m = 120$ . Если среди найденных пар максимальную сумму имеют несколько, то можно напечатать любую из них.

**Описание входных и выходных данных**

В первой строке входных данных задаётся количество чисел  $n$  ( $2 \leq n \leq 12\,000$ ). В каждой из последующих  $n$  строк записано одно целое положительное число, не превышающее 10 000.

В качестве результата программа должна напечатать элементы искомой пары. Если таких пар несколько, можно вывести любую из них. Гарантируется, что хотя бы одна такая пара в последовательности есть.

*Пример входных данных:*

6  
60  
140  
61  
100  
300  
59

*Пример выходных данных для приведённого выше примера входных данных:*

140 100

*Пояснение.* Из шести заданных чисел можно составить три пары, сумма элементов которых делится на  $m=120$ : 60+300, 140+100 и 61+59. Во второй и третьей из этих пар первый элемент больше второго, но во второй паре сумма больше.

Требуется написать эффективную по времени и памяти программу для решения описанной задачи.

Программа считается эффективной по времени, если при одновременном увеличении количества элементов последовательности  $n$  и параметра  $m$  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 4 килобайта 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 памяти, – 4 балла.

Максимальная оценка за правильную программу, возможно, неэффективную по памяти или время выполнения которой существенно зависит от величины  $m$ , – 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.

**Содержание верного ответа**

(допускаются иные формулировки ответа, не искажающие его смысла)

Сумма  $a_i$  и  $a_j$  делится на  $m$ , если сумма остатков этих чисел от деления на  $m$  равна 0 или  $m$ . Для каждого из остатков от деления на  $m$  среди уже просмотренных элементов будем хранить максимальное число, имеющее соответствующий остаток от деления на  $m$ . Для этого будем использовать массив  $r$  длиной  $m$ , изначально с элементами, равными 0. Все считанные значения при этом можно не хранить.

Очередное считанное число  $a$  будем рассматривать как возможный правый элемент искомой пары. Пусть остаток от деления  $a$  на  $m$  равен  $p$ . Тогда если  $r[m-p] > 0$ , то сумма  $a$  и  $r[m-p]$  делится на  $m$ , и при условии  $r[m-p] > a$  эта пара – кандидат для ответа. Если их сумма больше предыдущего ответа, то заменим его. При этом если остаток от деления  $a$  на  $m$  равен 0, то рассматривать надо пару  $a$  и  $r[0]$ .

По окончании обработки элемента  $a$  необходимо обновить элемент  $r[p]$  значением  $a$ , если  $a > r[p]$ .

Ниже приведена реализующая описанный алгоритм программа на языке Паскаль (использована версия PascalABC)

**Пример 1. Программа на языке Паскаль. Программа эффективна по времени и памяти**

```
const m = 120; {количество различных остатков}
var
  {хранение максимального значения для каждого из остатков}
  r: array[0..m-1] of integer;
  n, a, i, p, left, right: integer;
begin
  readln(n);
  {обнуление массива r}
  for i := 0 to m - 1 do
    r[i] := 0;
  {обнуление переменных для записи ответа}
  left := 0; right := 0;
  {ввод значений, поиск искомой пары}
  for i := 1 to n do
  begin
    readln(a); {считываем очередное значение}
    p := a mod m;
    if p = 0 then
    begin
      if (r[0] > a) and (r[0] + a > left + right) then
      begin
        left := r[0]; right := a {обновление ответа}
      end
    end
    else
    begin
      if (r[m - p] > a) and (r[m - p] + a > left + right) then
      begin
        left := r[m - p]; right := a {обновление ответа}
      end
    end;
    {обновление элемента r для соответствующего остатка}
    if a > r[p] then r[p] := a
  end;
  writeln(left, ' ', right)
end.
```

**Комментарии для проверяющего**

1. При таком решении хранится только очередной прочитанный элемент и информация о максимальных значениях, имеющих различные остатки от деления на  $m$  (на их хранение будет потрачено не более  $4m$  байт памяти, а на все переменные в целом – менее 4 килобайт). Таким образом, используемая память не зависит от длины последовательности. Время обработки очередного числа фиксировано, т.е. не зависит от длины последовательности и даже от величины  $m$ . Поэтому при увеличении длины последовательности в  $k$  раз время работы программы увеличивается не более чем в  $k$  раз. Таким образом, приведённая выше программа эффективна как по времени, так и по используемой памяти. Это решение оценивается 4 баллами.

Программа может не рассматривать отдельно случай  $p = 0$ , а учесть оба случая с помощью одной формулы:  $(m - p) \bmod m$ . Такой вариант реализации показан в примере 2 программы на языке Python. Может быть реализовано решение с заменой  $p = 0$  на  $p = m$ . Такая программа на языке C++ приведена ниже (пример 3).

Все подобные программы оцениваются, исходя из максимального балла – 4 (см. критерии).

2. Возможно решение, основанное на описанных идеях, однако предварительно сохраняющее элементы последовательности в массив или другие структуры данных. Такое решение эффективно по времени, но неэффективно по памяти. Оно оценивается, исходя из максимального балла – 3 (см. критерии). Кроме того, возможен неэффективный способ определения, какой именно остаток от деления нас интересует, например с помощью цикла, выполняющегося до  $m$  раз, или с помощью  $m$  условных операторов:

```
if p = 0 and a > r[0] then r[0] = a;
if p = 1 and a > r[1] then r[1] = a;
```

и т.д.

Такое решение работает в  $m$  раз дольше и оценивается, исходя из максимального балла – 3 (см. критерии).

3. Решение, неэффективное ни по времени, ни по памяти, запоминает входную последовательность в массиве, после чего явно перебирает все возможные пары. Такое решение оценивается, исходя из максимального балла – 2 (см. критерии)

**Пример 2. Программа на языке Python 3. Программа эффективна по времени и памяти**

```
m = 120
# создание массива для максимальных значений
# для каждого из остатков
r = [0] * m
# обнуление переменных для записи ответа
left = 0
right = 0
# ввод количества элементов
n = int(input())
# ввод значений, поиск искомой пары
for i in range(n):
  a = int(input())
  p = a % m;
  if r[(m - p) % m] > a and r[(m - p) % m] + a > left + right:
    #обновление ответа
    left = r[(m - p) % m]
    right = a;
  # обновление элемента r для соответствующего остатка
  if a > r[p]:
    r[p] = a
print(left, right)
```

**Пример 3. Программа на языке C++. Программа эффективна по времени и памяти**

```
#include <iostream>
using namespace std;

int main()
{
    int n, a, p, left, right;
    int r[120];
    int m = 120;
    cin >> n;
    //обнуление массива r
    for (int i = 0; i < m; ++i)
        r[i] = 0;
    //обнуление переменных для записи ответа
    left = 0; right = 0;
    // ввод значений, поиск искомой пары
    for (int i = 0; i < n; ++i)
    {
        cin >> a; //считываем очередное значение
        p = a % m;
        if (p == 0) p = m;
        if (r[m - p] > a && r[m - p] + a > left + right)
        {
            left = r[m - p]; right = a; //обновление ответа
        }
        // обновление элемента r для соответствующего остатка
        if (p < m)
        {
            if (a > r[p]) r[p] = a;
        }
        else if (a > r[0]) r[0] = a;
    }
    cout << left << ' ' << right;
}
```

Указания по оцениванию	Баллы
<p>Если в работе представлены две программы решения задачи, то каждая из них независимо оценивается по указанным ниже критериям, итоговой считается бóльшая из двух оценок. Описание алгоритма решения без программы оценивается в 0 баллов</p>	
<p>Программа правильно работает для любых входных данных произвольного размера при условии исправления в ней не более трёх синтаксических ошибок из приведённого ниже списка допустимых ошибок. Используемая память не зависит от количества прочитанных чисел, а время работы пропорционально этому количеству.</p> <p>Допускается наличие в тексте программы до трёх синтаксических ошибок одного из следующих видов:</p> <ol style="list-style-type: none"> <li>1) пропущен или неверно указан знак пунктуации;</li> <li>2) неверно написано, пропущено или написано лишнее зарезервированное слово языка программирования;</li> <li>3) не описана или неверно описана переменная;</li> <li>4) применяется операция, не допустимая для соответствующего типа данных.</li> </ol> <p>Если одна и та же ошибка встречается несколько раз, это считается за одну ошибку</p>	4
<p>Не выполнены условия, позволяющие поставить 4 балла.</p> <p>Программа работает правильно для любых входных данных произвольного размера при условии исправления в ней не более пяти синтаксических ошибок из приведённого в критериях на 4 балла списка и не более одной ошибки из приведённого ниже списка содержательных ошибок. Время работы пропорционально количеству введённых чисел, но может существенно зависеть от <i>m</i> (см. комментарий к эффективному решению задачи). Допускается наличие не более <b>одной</b> содержательной ошибки следующих видов:</p> <ol style="list-style-type: none"> <li>1) допущена ошибка при вводе данных (например, не считается значение <i>N</i>, или числа могут быть считаны, только если будут записаны в одной строке через пробел);</li> <li>2) неверная инициализация или её отсутствие там, где она необходима;</li> <li>3) используется неверный тип данных, при этом ошибка не является синтаксической;</li> <li>4) не более одного раза используется одна переменная (или константа) вместо другой, или не более одного раза используется один знак операции вместо другого, или не более одного раза используется одно зарезервированное слово языка программирования вместо другого, при этом ошибка не является синтаксической;</li> </ol>	3

<p>5) служебное слово else относится не к тому if, к какому следует;          6) отсутствует вывод ответа, или выводится значение не тех переменных;          7) выход за границу массива (в частности, при обращении к <math>m</math>-му элементу массива с индексами от 0 до <math>m-1</math>, даже если он существует, но не заполнен нужным значением);          8) не выполнен или неверно выполнен учёт элементов, остаток от деления которых на <math>m</math> равен 0.          3 балла также ставится за программу, в которой нет содержательных ошибок, но используемая память зависит от количества прочитанных чисел (например, входные данные запоминаются в массиве, контейнере STL в C++ или другой аналогичной структуре данных)</p>	
<p>Не выполнены условия, позволяющие поставить 3 или 4 балла.          Программа работает верно, эффективно по времени при условии исправления не более трёх содержательных ошибок, описанных в критериях на 3 балла и аналогичных им, и не более девяти синтаксических ошибок, указанных в критериях на 4 балла. При этом в программе могут быть опущены с помощью многоточия однотипные действия, связанные с рассмотрением каждого из остатков от деления на <math>m</math>.</p> <p>Не допускается выставление 2 баллов за программу, если в ней учитываются суммы вида <math>a[i]+a[i]</math> (в том числе в алгоритме без хранения элементов последовательности).          2 балла также ставится за корректное переборное решение, в котором все числа сохраняются в массиве (или другой аналогичной структуре) и рассматриваются все возможные пары. Пример фрагмента соответствующей программы на языке Паскаль:  <pre> left := 0; right := 0; for i := 1 to N - 1 do   for j := i + 1 to N do     if (a[i] &gt; a[j]) and ((a[i] + a[j]) mod m = 0) then       if a[i] + a[j] &gt; left + right then         begin           left := a[i];           right := a[j]         end; end; </pre>         В цикле реализации переборного алгоритма не допускаются выход индексов за границы массива, а также любые логические ошибки</p>	2

<p>Не выполнены условия, позволяющие поставить 2, 3 или 4 балла. При этом программа описывает в целом правильный алгоритм (эффективный или нет) и в ней присутствует не менее двух элементов решения из перечисленных ниже, возможно, реализованных с ошибками:</p> <ul style="list-style-type: none"> <li>• учитывается условие <math>a[i] &gt; a[j]</math>;</li> <li>• проверяется делимость суммы на <math>m</math>;</li> <li>• ищется пара с максимальной суммой.</li> </ul> <p>В случае, если в любом решении содержится не более одного из указанных элементов, программа оценивается в 0 баллов</p>	1
<p>Не выполнены критерии, позволяющие поставить 1, 2, 3 или 4 балла</p>	0
<i>Максимальный балл</i>	4

В соответствии с Порядком проведения государственной итоговой аттестации по образовательным программам среднего общего образования (приказ Минпросвещения России и Рособнадзора от 07.11.2018 № 190/1512, зарегистрирован Минюстом России 10.12.2018 № 52952)

«82. <...> По результатам первой и второй проверок эксперты независимо друг от друга выставляют баллы за каждый ответ на задания экзаменационной работы ЕГЭ с развернутым ответом. <...>

В случае существенного расхождения в баллах, выставленных двумя экспертами, назначается третья проверка. Существенное расхождение в баллах определено в критериях оценивания по соответствующему учебному предмету.

Эксперту, осуществляющему третью проверку, предоставляется информация о баллах, выставленных экспертами, ранее проверявшими экзаменационную работу».

Если расхождение составляет 2 или более балла за выполнение любого из заданий 24–27, то третий эксперт проверяет только те ответы на задания, которые вызвали столь существенное расхождение.

Единый государственный экзамен по ИНФОРМАТИКЕ и ИКТ

**Кодификатор**  
элементов содержания и требований к уровню  
подготовки выпускников образовательных  
организаций для проведения  
единого государственного экзамена  
по информатике и ИКТ

подготовлен Федеральным государственным бюджетным  
научным учреждением

«ФЕДЕРАЛЬНЫЙ ИНСТИТУТ ПЕДАГОГИЧЕСКИХ ИЗМЕРЕНИЙ»

**Кодификатор**  
**элементов содержания и требований к уровню подготовки выпускников**  
**образовательных организаций для проведения единого государственного**  
**экзамена по ИНФОРМАТИКЕ и ИКТ**

Кодификатор элементов содержания и требований к уровню подготовки выпускников образовательных организаций для проведения единого государственного экзамена по информатике и ИКТ (далее – кодификатор) является одним из документов, определяющих структуру и содержание контрольных измерительных материалов (далее – КИМ) единого государственного экзамена (далее – ЕГЭ). Он составлен на основе Федерального компонента государственных стандартов основного общего и среднего (полного) общего образования по информатике и ИКТ (базовый и профильный уровни) (приказ Минобрнауки России от 05.03.2004 № 1089).

В кодификатор не включены элементы содержания, выделенные курсивом в разделе стандарта «Обязательный минимум содержания основных образовательных программ»: данное содержание подлежит изучению, но не включено в раздел стандарта «Требования к уровню подготовки выпускников», т.е. не является объектом контроля. Также в кодификатор не включены те требования к уровню подготовки выпускников, достижение которых не может быть проверено в рамках единого государственного экзамена.

**Раздел 1. Перечень элементов содержания, проверяемых на едином государственном экзамене по информатике и ИКТ**

Перечень элементов содержания, проверяемых на едином государственном экзамене по информатике и ИКТ, составлен на основе раздела «Обязательный минимум содержания основных образовательных программ» Федерального компонента государственных стандартов среднего (полного) общего образования по информатике и ИКТ (базовый и профильный уровни).

В первом столбце указан код раздела, которому соответствуют крупные блоки содержания. Во втором столбце приводится код элемента содержания, для которого создаются проверочные задания. В третьем столбце приводится словесное описание контролируемого элемента содержания.

Код раз-дела	Код контролируемого элемента	Элементы содержания, проверяемые на ЕГЭ
1		<b>Информация и информационные процессы</b>
	1.1	Информация и ее кодирование
	1.1.1	Виды информационных процессов
	1.1.2	Процесс передачи информации, источник и приемник информации. Сигнал, кодирование и декодирование. Искажение информации
	1.1.3	Дискретное (цифровое) представление текстовой, графической, звуковой информации и видеoinформации. Единицы измерения количества информации
	1.1.4	Скорость передачи информации
	1.2	Системы, компоненты, состояние и взаимодействие компонентов. Информационное взаимодействие в системе, управление, обратная связь
	1.3	Моделирование
	1.3.1	Описание (информационная модель) реального объекта и процесса, соответствие описания объекту и целям описания. Схемы, таблицы, графики, формулы как описания
	1.3.2	Математические модели
	1.3.3	Использование сред имитационного моделирования (виртуальных лабораторий) для проведения компьютерного эксперимента в учебной деятельности
	1.4	Системы счисления
	1.4.1	Позиционные системы счисления
	1.4.2	Двоичное представление информации
	1.5	Логика и алгоритмы
	1.5.1	Высказывания, логические операции, кванторы, истинность высказывания
	1.5.2	Цепочки (конечные последовательности), деревья, списки, графы, матрицы (массивы), псевдослучайные последовательности
	1.5.3	Индуктивное определение объектов
	1.5.4	Вычислимые функции, полнота формализации понятия вычислимости, универсальная вычислимая функция
	1.5.5	Кодирование с исправлением ошибок
1.5.6	Сортировка	
1.6	Элементы теории алгоритмов	

	1.6.1	Формализация понятия алгоритма
	1.6.2	Вычислимость. Эквивалентность алгоритмических моделей
	1.6.3	Построение алгоритмов и практические вычисления
	1.7	Языки программирования
	1.7.1	Типы данных
	1.7.2	Основные конструкции языка программирования. Система программирования
2	1.7.3	Основные этапы разработки программ. Разбиение задачи на подзадачи
		<b>Информационная деятельность человека</b>
	2.1	Профессиональная информационная деятельность. Информационные ресурсы
	2.2	Экономика информационной сферы
3	2.3	Информационная этика и право, информационная безопасность
		<b>Средства ИКТ</b>
	3.1	Архитектура компьютеров и компьютерных сетей
	3.1.1	Программная и аппаратная организация компьютеров и компьютерных систем. Виды программного обеспечения
	3.1.2	Операционные системы. Понятие о системном администрировании
	3.1.3	Безопасность, гигиена, эргономика, ресурсосбережение, технологические требования при эксплуатации компьютерного рабочего места
	3.2	Технологии создания и обработки текстовой информации
	3.2.1	Понятие о настольных издательских системах. Создание компьютерных публикаций
	3.2.2	Использование готовых и создание собственных шаблонов. Использование систем проверки орфографии и грамматики. Тезаурусы. Использование систем двуязычного перевода и электронных словарей
	3.2.3	Использование специализированных средств редактирования математических текстов и графического представления математических объектов
	3.2.4	Использование систем распознавания текстов
	3.3	Технология создания и обработки графической и мультимедийной информации
	3.3.1	Форматы графических и звуковых объектов
	3.3.2	Ввод и обработка графических объектов
3.3.3	Ввод и обработка звуковых объектов	
3.4	Обработка числовой информации	

<b>3.4.1</b>	Математическая обработка статистических данных
<b>3.4.2</b>	Использование динамических (электронных) таблиц для выполнения учебных заданий из различных предметных областей
<b>3.4.3</b>	Использование инструментов решения статистических и расчетно-графических задач
<b>3.5</b>	Технологии поиска и хранения информации
<b>3.5.1</b>	Системы управления базами данных. Организация баз данных
<b>3.5.2</b>	Использование инструментов поисковых систем (формирование запросов)
<b>3.6</b>	Телекоммуникационные технологии
<b>3.6.1</b>	Специальное программное обеспечение средств телекоммуникационных технологий
<b>3.6.2</b>	Инструменты создания информационных объектов для Интернета
<b>3.7</b>	Технологии управления, планирования и организации деятельности человека

**Раздел 2. Перечень требований к уровню подготовки выпускников, достижение которого проверяется на едином государственном экзамене по информатике и ИКТ**

Перечень требований к уровню подготовки выпускников, достижение которого проверяется на едином государственном экзамене по информатике и ИКТ, составлен с учетом сформулированных в образовательном стандарте целей изучения предмета, а также на основе раздела «Требования к уровню подготовки выпускников» Федерального компонента государственных стандартов среднего (полного) общего образования по информатике и ИКТ (базовый и профильный уровни).

В первом столбце даны коды требований; во втором столбце – требования к уровню подготовки выпускников, достижение которого проверяется на ЕГЭ.

<b>Код требований</b>	<b>Проверяемые умения или способы действий</b>
<b>1</b>	<b>ЗНАТЬ/ПОНИМАТЬ/УМЕТЬ:</b>
<b>1.1</b>	Моделировать объекты, системы и процессы
<b>1.1.1</b>	Проводить вычисления в электронных таблицах
<b>1.1.2</b>	Представлять и анализировать табличную информацию в виде графиков и диаграмм
<b>1.1.3</b>	Строить информационные модели объектов, систем и процессов в виде алгоритмов
<b>1.1.4</b>	Читать и отлаживать программы на языке программирования
<b>1.1.5</b>	Создавать программы на языке программирования по их описанию
<b>1.1.6</b>	Строить модели объектов, систем и процессов в виде таблицы истинности для логического высказывания
<b>1.1.7</b>	Вычислять логическое значение сложного высказывания по известным значениям элементарных высказываний
<b>1.2</b>	Интерпретировать результаты моделирования
<b>1.2.1</b>	Использовать готовые модели, оценивать их соответствие реальному объекту и целям моделирования
<b>1.2.2</b>	Интерпретировать результаты, получаемые в ходе моделирования реальных процессов
<b>1.3</b>	Оценивать числовые параметры информационных объектов и процессов
<b>1.3.1</b>	Оценивать объем памяти, необходимый для хранения информации
<b>1.3.2</b>	Оценивать скорость передачи и обработки информации
<b>2</b>	<b>ИСПОЛЬЗОВАТЬ ПРИОБРЕТЕННЫЕ ЗНАНИЯ И УМЕНИЯ В ПРАКТИЧЕСКОЙ ДЕЯТЕЛЬНОСТИ И ПОВСЕДНЕВНОЙ ЖИЗНИ:</b>
<b>2.1</b>	Осуществлять поиск и отбор информации
<b>2.2</b>	Создавать и использовать структуры хранения данных
<b>2.3</b>	Работать с распространенными автоматизированными информационными системами
<b>2.4</b>	Готовить и проводить выступления, участвовать в коллективном обсуждении, фиксировать его ход и результаты с использованием современных программных и аппаратных средств коммуникаций
<b>2.5</b>	Проводить статистическую обработку данных с помощью компьютера
<b>2.6</b>	Выполнять требования техники безопасности, гигиены, эргономики и ресурсосбережения при работе со средствами информатизации

Возможные алгоритмические задачи для подраздела 1.1 перечня требований к уровню подготовки выпускников, достижение которых проверяется на едином государственном экзамене по информатике и ИКТ.

- Нахождение минимума и максимума двух, трех, четырех данных чисел без использования массивов и циклов.
- Нахождение всех корней заданного квадратного уравнения.
- Запись натурального числа в позиционной системе с основанием, меньшим или равным 10. Обработка и преобразование такой записи числа.
- Нахождение сумм, произведений элементов данной конечной числовой последовательности (или массива).
- Использование цикла для решения простых переборных задач (поиск наименьшего простого делителя данного натурального числа, проверка числа на простоту и т.д.).
- Заполнение элементов одномерного и двумерного массивов по заданным правилам.
- Операции с элементами массива. Линейный поиск элемента. Вставка и удаление элементов в массиве. Перестановка элементов данного массива в обратном порядке. Суммирование элементов массива. Проверка соответствия элементов массива некоторому условию.
- Нахождение второго по величине (второго максимального или второго минимального) значения в данном массиве за однократный просмотр массива.
- Нахождение минимального (максимального) значения в данном массиве и количества элементов, равных ему, за однократный просмотр массива.
- Операции с элементами массива, отобранных по некоторому условию (например, нахождение минимального четного элемента в массиве, нахождение количества и суммы всех четных элементов в массиве).
- Сортировка массива.
- Слияние двух упорядоченных массивов в один без использования сортировки.
- Обработка отдельных символов данной строки. Подсчет частоты появления символа в строке.
- Работа с подстроками данной строки с разбиением на слова по пробельным символам. Поиск подстроки внутри данной строки, замена найденной подстроки на другую строку.

Единый государственный экзамен по ИНФОРМАТИКЕ и ИКТ

**Спецификация**  
контрольных измерительных материалов  
для проведения в 2020 году  
единого государственного экзамена  
по информатике и ИКТ

подготовлена Федеральным государственным бюджетным  
научным учреждением

«ФЕДЕРАЛЬНЫЙ ИНСТИТУТ ПЕДАГОГИЧЕСКИХ ИЗМЕРЕНИЙ»

**СПЕЦИФИКАЦИЯ**  
контрольных измерительных материалов  
единого государственного экзамена 2020 года  
по информатике и ИКТ

**1. Назначение контрольных измерительных материалов (КИМ) ЕГЭ**

Единый государственный экзамен (ЕГЭ) представляет собой форму государственной итоговой аттестации, проводимой в целях определения соответствия результатов освоения обучающимися основных образовательных программ среднего общего образования соответствующим требованиям федерального государственного образовательного стандарта или образовательного стандарта. Для указанных целей используются контрольные измерительные материалы (КИМ), представляющие собой комплексы заданий стандартизированной формы.

ЕГЭ проводится в соответствии с Федеральным законом «Об образовании в Российской Федерации» от 29.12.2012 № 273-ФЗ и Порядком проведения государственной итоговой аттестации по образовательным программам среднего общего образования, утверждённого приказом Минпросвещения России и Рособразнадзора от 07.11.2018 № 190/1512.

**2. Документы, определяющие содержание КИМ ЕГЭ**

Содержание КИМ определяется на основе Федерального компонента государственного стандарта среднего (полного) общего образования, базовый и профильный уровни (приказ Минобрнауки России от 05.03.2004 № 1089)

**3. Подходы к отбору содержания, разработке структуры КИМ ЕГЭ**

Содержание заданий разработано по основным темам курса информатики и ИКТ, объединённых в следующие тематические блоки: «Информация и ее кодирование», «Моделирование и компьютерный эксперимент», «Системы счисления», «Логика и алгоритмы», «Элементы теории алгоритмов», «Программирование», «Архитектура компьютеров и компьютерных сетей», «Обработка числовой информации», «Технологии поиска и хранения информации».

Содержанием экзаменационной работы охватывается основное содержание курса информатики и ИКТ, важнейшие его темы, наиболее значимый в них материал, однозначно трактуемый в большинстве преподаваемых в школе вариантов курса информатики и ИКТ.

Работа содержит как задания базового уровня сложности, проверяющие знания и умения, предусмотренные стандартом базового уровня, так и задания повышенного и высокого уровней сложности, проверяющие знания

и умения, предусмотренные стандартом профильного уровня. Количество заданий в варианте КИМ должно, с одной стороны, обеспечить всестороннюю проверку знаний и умений выпускников, приобретенных за весь период обучения по предмету, и, с другой стороны, соответствовать критериям сложности, устойчивости результатов, надежности измерения. С этой целью в КИМ используются задания двух типов: с кратким ответом и развернутым ответом. Структура экзаменационной работы обеспечивает оптимальный баланс заданий разных типов и разновидностей, трех уровней сложности, проверяющих знания и умения на трех различных уровнях: воспроизведения, применения в стандартной ситуации, применения в новой ситуации. Содержание экзаменационной работы отражает значительную часть содержания предмета. Все это обеспечивает валидность результатов экзамена и надежность измерения.

#### 4. Структура КИМ ЕГЭ

Каждый вариант экзаменационной работы состоит из двух частей и включает в себя 27 заданий, различающихся формой и уровнем сложности.

Часть 1 содержит 23 задания с кратким ответом.

В экзаменационной работе предложены следующие разновидности заданий с кратким ответом:

- задания на вычисление определенной величины;
- задания на установление правильной последовательности, представленной в виде строки символов по определенному алгоритму.

Ответ на задания части 1 дается соответствующей записью в виде натурального числа или последовательности символов (букв или цифр), записанных без пробелов и других разделителей.

Часть 2 содержит 4 задания с развернутым ответом.

Часть 1 содержит 23 задания базового, повышенного и высокого уровней сложности. В этой части собраны задания с кратким ответом, подразумевающие самостоятельное формулирование и запись ответа в виде числа или последовательности символов. Задания проверяют материал всех тематических блоков. В части 1 12 заданий относятся к базовому уровню, 10 заданий – к повышенному уровню сложности, 1 задание – к высокому уровню сложности.

Часть 2 содержит 4 задания, первое из которых повышенного уровня сложности, остальные 3 задания высокого уровня сложности. Задания этой части подразумевают запись развернутого ответа в произвольной форме.

Задания части 2 направлены на проверку сформированности важнейших умений записи и анализа алгоритмов. Эти умения проверяются на повышенном и высоком уровнях сложности. Также на высоком уровне сложности проверяются умения по теме «Технология программирования».

Распределение заданий по частям экзаменационной работы представлено в таблице 1.

Таблица 1  
Распределение заданий по частям экзаменационной работы

Часть работы	Количество заданий	Максимальный первичный балл	Процент максимального первичного балла за выполнение заданий данной части от максимального первичного балла за всю работу, равного 35	Тип заданий
Часть 1	23	23	66	С кратким ответом
Часть 2	4	12	34	С развернутым ответом
Итого	27	35	100	

#### 5. Распределение заданий КИМ ЕГЭ по содержанию, видам умений и способам действий

Отбор содержания, подлежащего проверке в КИМ ЕГЭ 2020 г., осуществляется на основе Федерального компонента государственного стандарта среднего (полного) общего образования (базовый и профильный уровни). Распределение заданий по разделам курса информатики и ИКТ представлено в таблице 2.

Таблица 2  
Распределение заданий экзаменационной работы по содержательным разделам курса информатики и ИКТ

№	Содержательные разделы	Количество заданий	Максимальный первичный балл	Процент максимального первичного балла за выполнение заданий данного раздела от максимального первичного балла за всю работу, равного 35
1	Информация и ее кодирование	4	4	11
2	Моделирование и компьютерный эксперимент	2	2	6
3	Системы счисления	2	2	6
4	Логика и алгоритмы	6	8	23
5	Элементы теории алгоритмов	5	6	17
6	Программирование	4	9	25
7	Архитектура компьютеров и компьютерных сетей	1	1	3
8	Обработка числовой информации	1	1	3
9	Технологии поиска и хранения информации	2	2	6
	Итого	27	35	100

В КИМ ЕГЭ по информатике и ИКТ не включены задания, требующие простого воспроизведения знания терминов, понятий, величин, правил (такие задания слишком просты для выполнения). При выполнении любого из заданий КИМ от экзаменуемого требуется решить тематическую задачу: либо прямо использовать известное правило, алгоритм, умение, либо выбрать из общего количества изученных понятий и алгоритмов наиболее подходящее и применить его в известной или новой ситуации.

Знание теоретического материала проверяется косвенно через понимание используемой терминологии, взаимосвязей основных понятий, размерностей единиц и т.д. при выполнении экзаменуемыми практических заданий по различным темам предмета. Таким образом, в КИМ по информатике и ИКТ проверяется освоение теоретического материала из разделов:

- единицы измерения информации;
- принципы кодирования;
- системы счисления;
- моделирование;
- понятие алгоритма, его свойств, способов записи;
- основные алгоритмические конструкции;
- основные понятия, используемые в информационных и коммуникационных технологиях.

Экзаменационная работа содержит одно задание, требующее прямо применить изученное правило, формулу, алгоритм. Это задание (1) отмечено как задание на воспроизведение знаний и умений.

Материал на проверку сформированности *умений применять свои знания в стандартной ситуации* входит в обе части экзаменационной работы. Это следующие умения:

- анализировать однозначность двоичного кода;
- формировать для логической функции таблицу истинности и логическую схему;
- оперировать массивами данных;
- подсчитать информационный объем сообщения;
- искать кратчайший путь в графе, осуществлять обход графа;
- осуществлять перевод из одной системы счисления в другую;
- использовать стандартные алгоритмические конструкции при программировании;
- формально исполнять алгоритмы, записанные на естественных и алгоритмических языках, в том числе на языках программирования;
- определять мощность адресного пространства компьютерной сети по маске подсети в протоколе ТСР/ІР;
- оценить результат работы известного программного обеспечения;
- формулировать запросы к базам данных и поисковым системам.

Материал на проверку сформированности *умений применять свои знания в новой ситуации* также входит в обе части экзаменационной работы. Это следующие сложные умения:

- анализировать обстановку исполнителя алгоритма;
- определять основание системы счисления по свойствам записи чисел;
- описывать свойства двоичной последовательности по алгоритму ее построения;
- осуществлять преобразования логических выражений;
- моделировать результаты поиска в сети Интернет;
- анализировать результат исполнения алгоритма;
- анализировать текст программы с точки зрения соответствия записанного алгоритма поставленной задаче и изменять его в соответствии с заданием;
- умение построить дерево игры по заданному алгоритму и обосновать выигрышную стратегию;
- реализовывать сложный алгоритм с использованием современных систем программирования.

Каждое задание экзаменационной работы характеризуется не только проверяемым содержанием, но и проверяемыми умениями. Кодификатор определяет две группы требований к уровню подготовки выпускников: с одной стороны, знать/понимать/уметь и, с другой стороны, использовать приобретенные знания и умения в практической деятельности и повседневной жизни.

При том что стандарт образования по информатике и ИКТ содержит достаточно много требований к использованию приобретенных знаний и умений в практической жизни, используемая стандартизированная бланковая технология единого государственного экзамена не позволяет проверить выполнение этих требований в полном объеме. В работе всего 3 таких задания, они расположены в части 1 работы. Их выполнение дает менее 10% первичных баллов. Остальные 90% первичных баллов экзаменуемый может получить за счет реализации умений оперировать с теоретическим материалом предмета информатики и ИКТ. В таблице 3 характеризуется распределение заданий с точки зрения проверяемых умений в каждой части работы.

Таблица 3

Распределение заданий экзаменационной работы по видам проверяемых умений и способам действий

Основные умения и способы действий	Количество заданий		
	Вся работа	Часть 1	Часть 2
<b>1. Требования: «Знать/понимать/уметь»</b>	<b>24</b>	<b>20</b>	<b>4</b>
Моделирование объектов, систем и процессов	16	12	4
Интерпретация результатов моделирования	4	4	0
Определение количественных параметров информационных процессов	4	4	0
<b>2. Требования: «Использовать приобретенные знания и умения в практической деятельности и повседневной жизни»</b>	<b>3</b>	<b>3</b>	<b>0</b>
Осуществлять поиск и отбор информации	1	1	–
Создавать и использовать структуры хранения данных	1	1	–
Работать с распространенными автоматизированными информационными системами	1	1	–
Итого	27	23	4

### 6. Распределение заданий КИМ ЕГЭ по уровню сложности

Часть 1 экзаменационной работы содержит 12 заданий базового уровня сложности, 10 заданий повышенного уровня и 1 задание высокого уровня сложности.

Задания части 2 относятся к повышенному (1 задание) и высокому уровням. Предполагаемый процент выполнения заданий базового уровня – 60–90. Предполагаемый процент выполнения заданий повышенного уровня – 40–60. Предполагаемый процент выполнения заданий высокого уровня – менее 40.

Для оценки достижения выпускником базового уровня используются задания с кратким ответом. Достижение повышенного уровня подготовки проверяется с помощью заданий с кратким и развернутым ответами. Для проверки достижения высокого уровня подготовки в экзаменационной работе используются задания с кратким и развернутым ответами. Распределение заданий по уровням сложности представлено в таблице 4.

Таблица 4

Распределение заданий по уровням сложности

Уровень сложности заданий	Количество заданий	Максимальный первичный балл	Процент максимального первичного балла за выполнение заданий данного уровня сложности от максимального первичного балла за всю работу, равного 35
Базовый	12	12	34
Повышенный	11	13	37
Высокий	4	10	29
Итого	27	35	100

Внутри каждой из двух частей работы задания расположены по принципу нарастающей сложности. Сначала идут задания базового уровня; затем – повышенного; затем – высокого. Задания одного уровня сложности расположены с учетом вида проверяемой деятельности и последовательности расположения тем в кодификаторе содержания.

### 7. Продолжительность ЕГЭ по информатике и ИКТ

На выполнение экзаменационной работы отводится 3 часа 55 минут (235 минут). На выполнение заданий части 1 рекомендуется отводить 1,5 часа (90 минут). Остальное время рекомендуется отводить на выполнение заданий части 2.

### 8. Дополнительные материалы и оборудование

Дополнительные материалы и оборудование не используются.

### 9. Система оценивания выполнения отдельных заданий и экзаменационной работы в целом

Задания КИМ оцениваются разным количеством баллов в зависимости от их типа.

Выполнение каждого задания части 1 оценивается в 1 балл. Задание части 1 считается выполненным, если экзаменуемый дал ответ, соответствующий коду верного ответа. За выполнение каждого задания присваивается (в дихотомической системе оценивания) либо 0 баллов («задание не выполнено»), либо 1 балл («задание выполнено»). Ответы на задания части 1 автоматически обрабатываются после сканирования бланков ответов № 1.

Максимальное количество первичных баллов, которое можно получить за выполнение заданий части 1, – 23.

Выполнение заданий части 2 оценивается от 0 до 4 баллов. Ответы на задания части 2 проверяются и оцениваются экспертами. Максимальное количество баллов, которое можно получить за выполнение заданий части 2, – 12.

Максимальный первичный балл – 35.

В соответствии с Порядком проведения государственной итоговой аттестации по образовательным программам среднего общего образования (приказ Минпросвещения России и Рособрнадзора от 07.11.2018 № 190/1512 зарегистрирован Минюстом России 10.12.2018 № 52952)

«82. <...> По результатам первой и второй проверок эксперты независимо друг от друга выставляют баллы за каждый ответ на задания экзаменационной работы ЕГЭ с развернутым ответом <...>

В случае существенного расхождения в баллах, выставленных двумя экспертами, назначается третья проверка. Существенное расхождение в баллах определено в критериях оценивания по соответствующему учебному предмету.

Эксперту, осуществляющему третью проверку, предоставляется информация о баллах, выставленных экспертами, ранее проверявшими экзаменационную работу».

Если расхождение составляет 2 или более балла за выполнение любого из заданий 24–27, то третий эксперт проверяет ответы только те ответы на задания, которые вызвали столь существенное расхождение.

На основе результатов выполнения всех заданий работы определяются первичные баллы, которые затем переводятся в тестовые по 100-балльной шкале.

#### 10. Изменения в КИМ 2020 года по сравнению с КИМ 2019 года

Изменения структуры КИМ отсутствуют.

Приложение

#### Обобщенный план варианта КИМ ЕГЭ 2020 года по ИНФОРМАТИКЕ и ИКТ

Уровни сложности заданий: Б – базовый; П – повышенный; В – высокий.

№	Проверяемые элементы содержания	Коды проверяемых элементов содержания (по кодификатору)	Коды проверяемых требований к уровню подготовки (по кодификатору)	Уровень сложности задания	Макс. балл за выполнение задания	Примерное время выполнения задания (мин.)
<b>Часть 1</b>						
1	Знание о системах счисления и двоичном представлении информации в памяти компьютера	1.4.2	1.3	Б	1	1
2	Умение строить таблицы истинности и логические схемы	1.5.1	1.1.6	Б	1	3
3	Умение представлять и считывать данные в разных типах информационных моделей (схемы, карты, таблицы, графики и формулы)	1.3.1	1.2.2	Б	1	3
4	Знание о файловой системе организации данных или о технологии хранения, поиска и сортировки информации в базах данных	3.1.2/ 3.5.1	2.1/ 2.2	Б	1	3
5	Умение кодировать и декодировать информацию	1.1.2	1.2.2	Б	1	2
6	Формальное исполнение алгоритма, записанного на естественном языке или умение создавать линейный алгоритм для формального исполнителя с ограниченным набором команд	1.6.1/ 1.6.3	1.1.3	Б	1	4
7	Знание технологии обработки информации в электронных таблицах и методов визуализации данных с помощью диаграмм и графиков	3.4.1/ 3.4.3	1.1.1/ 1.1.2	Б	1	3
8	Знание основных конструкций языка программирования, понятия переменной, оператора присваивания	1.7.2	1.1.4	Б	1	3
9	Умение определять скорость передачи информации при заданной пропускной способности канала, объем памяти, необходимый для хранения звуковой и графической информации	1.1.4/ 3.3.1	1.3.1/ 1.3.2	Б	1	5
10	Знание о методах измерения количества информации	1.1.3	1.3.1	Б	1	4
11	Умение исполнить рекурсивный алгоритм	1.5.3	1.1.3	Б	1	5

12	Знание базовых принципов организации и функционирования компьютерных сетей, адресации в сети	3.1.1	2.3	Б	1	2
13	Умение подсчитывать информационный объем сообщения	1.1.3	1.3.1	П	1	3
14	Умение исполнить алгоритм для конкретного исполнителя с фиксированным набором команд	1.6.2	1.2.2	П	1	6
15	Умение представлять и считывать данные в разных типах информационных моделей (схемы, карты, таблицы, графики и формулы)	1.3.1	1.2.1	П	1	3
16	Знание позиционных систем счисления	1.4.1	1.1.3	П	1	2
17	Умение осуществлять поиск информации в сети Интернет	3.5.2	2.1	П	1	2
18	Знание основных понятий и законов математической логики	1.5.1	1.1.7	П	1	3
19	Работа с массивами (заполнение, считывание, поиск, сортировка, массовые операции и др.)	1.5.2/ 1.5.6	1.1.4	П	1	5
20	Анализ алгоритма, содержащего цикл и ветвление	1.6.1	1.1.4	П	1	5
21	Умение анализировать программу, использующую процедуры и функции	1.7.2	1.1.4	П	1	6
22	Умение анализировать результат исполнения алгоритма	1.6.2	1.1.3	П	1	7
23	Умение строить и преобразовывать логические выражения	1.5.1	1.1.7	В	1	10
<b>Часть 2</b>						
24	Умение прочесть фрагмент программы на языке программирования и исправить допущенные ошибки	1.7.2	1.1.4	П	3	30
25	Умение составить алгоритм и записать его в виде простой программы (10–15 строк) на языке программирования	1.6.3	1.1.5	В	2	30
26	Умение построить дерево игры по заданному алгоритму и обосновать выигрышную стратегию	1.5.2	1.1.3	В	3	30
27	Умение создавать собственные программы (30–50 строк) для решения задач средней сложности	1.7.3	1.1.5	В	4	55
<p>Всего заданий – <b>27</b>; из них  по типу заданий: с кратким ответом – <b>23</b>; с развернутым ответом – <b>4</b>;  по уровню сложности: Б – <b>12</b>, П – <b>11</b>, В – <b>4</b>.  Максимальный первичный балл за работу – <b>35</b>.  Общее время выполнения работы – <b>235 мин.</b></p>						