Редакция от 20 ноя 2019

Как подготовить выпускников к ЕГЭ по информатике

Вера Савостина, учитель химии и информатики ГБОУ СОШ города Москвы «Школа № 64»

Анастасия Ветрова, эксперт Системы Завуч

Читайте в рекомендации, какие типичные ошибки допускают выпускники на ЕГЭ по информатике в первой и второй частях экзамена и как их избежать.

Как пройдет ЕГЭ по информатике в 2020 году

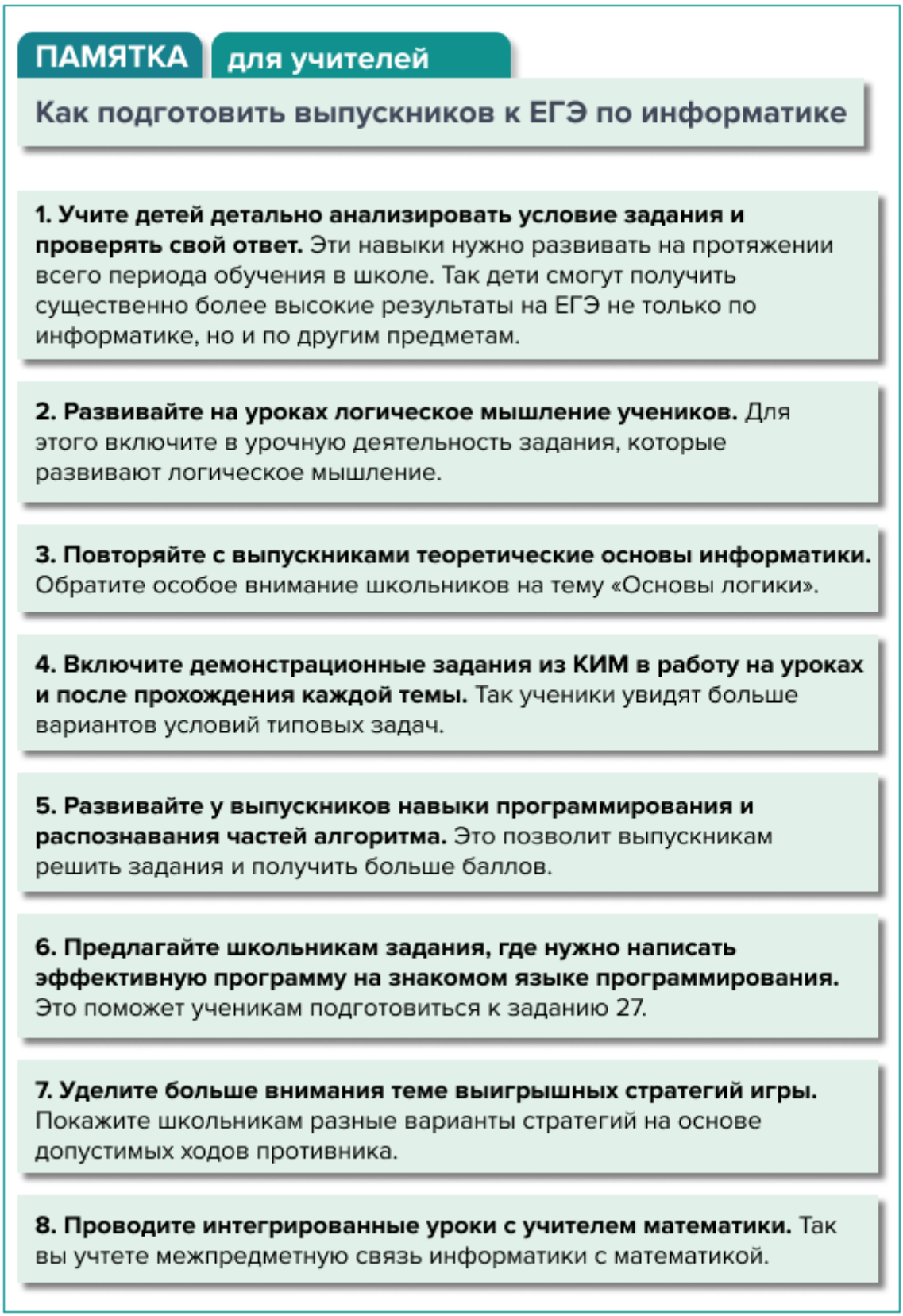
Изменений в структуре КИМ ЕГЭ по информатике в 2020 году нет. Поэтому для подготовки к экзамену можно использовать варианты ЕГЭ прошлых лет.

Экзамен по информатике состоит из 27 заданий. Они делятся на две части: 23 задания в первой части и 4 задания во второй.

Решение задач из первой части позволит набрать 23 первичных балла – по одному баллу за выполненное задание. Решение задач второй части добавит 12 первичных баллов. Максимум первичных баллов, которые выпускник получит за решение всех заданий, – 35.

При подготовке к ЕГЭ по информатике педагогу необходимо уделить особое внимание решению задач с развернутым ответом. Выпускник наберет больше баллов, если правильно решит эти задачи. Раздайте учителям памятки, которые помогут подготовить учеников к экзамену по информатике.

Памятка для учителей «Как подготовить выпускников к ЕГЭ по информатике»



Скачать

Какие ошибки выпускники допускают в первой части

Первая часть состоит из заданий с кратким ответом. Это задания № 1–23. За каждое задание выпускник может получить максимум один балл.

Порекомендуйте учителю разобрать со школьниками типичные ошибки. Это поможет выпускникам понять, где чаще всего ошибались ученики предыдущих лет, и самим не допустить этих ошибок.

Задание 2

Задание проверяет, как выпускники умеют строить таблицы истинности и логические схемы.

Пример

Пример задания 2

Миша заполнял таблицу истинности функции (x /\ ¬y) \/ (x≡z) \/ ¬w, но успел заполнить лишь фрагмент из трех **различных** ее 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 ¬x \/ 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: xwzy.

Частые ошибки. Типичная ошибка в таких заданиях встречается из-за того, что выпускники неправильно выстраивают последовательность столбцов. Это приводит к появлению идентичных строк, которые противоречат определению таблицы истинности и условию задачи. Также школьники путают обозначения конъюнкции и дизъюнкции.

Как избежать. Чтобы избежать ошибок в подобных заданиях, учителю нужно развивать у выпускников метапредметные навыки смыслового чтения и самостоятельной проверки

ответа. Порекомендуйте педагогу обратить внимание школьников на форму записи логических выражений в заданиях ЕГЭ. Это позволит выпускникам повторить правила записи логических операций и не запутаться на экзамене.

Чтобы успешно выполнять задания подобного типа, выпускнику надо повторить:

- основные законы алгебры логики: логическая сумма, логическое умножение, импликация, эквивалентность;
- правила записи импликации;
- условные обозначения логических операций;
- правила построения таблиц истинности;
- порядок выполнения логических операций;
- формулы де Моргана, чтобы упростить логические выражения.

Задание 9

Задания подобного типа проверяют, как выпускники умеют определять скорость передачи информации при определенной пропускной способности канала, необходимый объем памяти для хранения графической и звуковой информации.

Пример

Пример задания 9

Для хранения произвольного растрового изображения размером 128×320 пикселей отведено 40 Кбайт памяти без учета 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: 256.

Частые ошибки. Ученики допускают в задании содержательные ошибки. Выпускники путают количество битов, которое необходимо для хранения целочисленных значений из заданного диапазона, с количеством этих значений. Проблема возникает из-за того, что школьники заучивают комбинаторную формулу, но не понимают ее.

Как избежать. Проверьте, включил ли учитель-предметник в план подготовки к экзамену темы «Алфавитный подход к измерению информации» и «Кодирование сообщений словами фиксированной длины над заданным алфавитом». Учителю нужно математически строго излагать темы, четко формулировать определения, доказать формулы и факты, которые применяются в решении задач подобного типа. Также нужно добиться от учеников понимания комбинаторной формулы зависимости количества возможных кодовых слов от мощности алфавита и длины слова.

Чтобы успешно выполнять задания подобного типа, выпускнику надо повторить:

- какую формулу использовать, чтобы вычислить количество памяти для хранения растрового изображения;
- как вычислить количество пикселей в растровом изображении;

- что такое глубина кодирования цвета и как ее найти;
- как переводить Мбайты и Кбайты в биты.

Задание 11

Задание проверяет, как выпускники умеют исполнять рекурсивные алгоритмы.

Пример

Пример задания 11

Ниже на пяти языках программирования записан рекурсивный алгоритм F.

Бейсик

```
SUB F(n)
  PRINT n,
  IF n >= 3 THEN
    F(n \ 2)
    F(n - 1)
  END IF
END SUB
```

Python

```
def F(n):
    print(n, end='')
    if n >= 3:
        F(n // 2)
        F(n - 1)
```

Алгоритмический язык

```
алг F(цел n)
нач
  вывод n
  если n >= 3 то
    F(div(n, 2))
    F(n - 1)
  все
кон
```

Паскаль

```
procedure F(n: integer);
begin
  write(n);
  if n >= 3 then
  begin
    F(n div 2);
    F(n - 1)
  end
end;
```

C++

```
void F(int n) {
  std::cout << n;
  if (n >= 3) {
    F(n / 2);
    F(n - 1);
  }
}
```

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

Ответ: 5242312.

Частые ошибки. Ученики не могут построить верную последовательность сложных повторяющихся вызовов.

Как избежать. Учителю необходимо развить у выпускников умение исполнить алгоритм с простым ветвлением и вызовом функции, который записан на языке программирования. Также развивать метапредметное умение анализировать информацию.

Чтобы успешно выполнять задания подобного типа, выпускнику надо повторить:

- что такое рекурсия;
- как определить рекурсию;

- как запрограммировать рекурсию;
- для чего нужна рекурсия.

Задание 12

Задание проверяет, как выпускники знают базовые принципы организации и функционирования сетей, адресации в сети.

Пример

Пример задания 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 четырех байтов, причем 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: 192.

Частые ошибки. При решении подобных задач школьники часто невнимательно выполняют арифметические вычисления при переводе элементов IP-адреса из десятичной системы счисления в двоичную. Они не понимают, как формируется маска сети, и не умеют анализировать простые математические операции, к которым относится поразрядное логическое умножение.

Как избежать. Чтобы справиться с подобными заданиями, выпускник должен повторить:

- понятие адреса документа в интернете;
- правила записи адреса документа в сети;
- принципы адресации сети;
- запись IP-адреса в двоичной системе счисления;
- правила перевода IP-адреса из двоичной системы счисления в десятичную.

Как помочь выпускникам избежать ошибок во второй части

При подготовке к экзамену учителю надо обратить внимание выпускников на критерии оценивания заданий. Они дают представление о требованиях и возможных ошибках учеников. Также педагогу надо развивать у выпускников метапредметные навыки: анализ условия задания и способность к самопроверке.

Задание 24

Выпускникам нужно уметь исполнять и понимать алгоритм, записанный на языке программирования. В этом задании даются два–три вопроса. Если ученик сможет ответить на все вопросы, то сможет найти ошибки в коде программы.

Пример

Пример задания 24

На обработку поступает натуральное число, не превышающее 109 . Нужно написать программу, которая выводит на экран минимальную четную цифру этого числа. Если в числе нет четных цифр, требуется на экран вывести «NO». Программист написал программу неправильно. Ниже эта программа для вашего удобства приведена на пяти языках программирования.

Бейсик

```
DIM N, DIGIT, MINDIGIT AS LONG
INPUT N
MINDIGIT = N MOD 10
WHILE N > 0
  DIGIT = N MOD 10
  IF DIGIT MOD 2 = 0 THEN
    IF DIGIT < MINDIGIT THEN
      MINDIGIT = DIGIT
    END IF
  END IF
  N = N \ 10
WEND
IF MINDIGIT = 0 THEN
  PRINT "NO"
ELSE
  PRINT MINDIGIT
END IF
```

Python

```
N = int(input())
minDigit = N % 10
while N > 0:
  digit = N % 10
  if digit % 2 == 0:
    if digit < minDigit:
      minDigit = digit
  N = N // 10
if minDigit == 0:
  print("NO")
else:
  print(minDigit)
```

Алгоритмический язык

```
алг
нач
  цел N, digit, minDigit
  ввод N
  minDigit := mod(N,10)
  нц пока N > 0
    digit := mod(N,10)
    если mod(digit, 2) = 0 то
      если digit < minDigit то
        minDigit := digit
      все
    все
    N := div(N,10)
  кц
  если minDigit = 0 то
    вывод "NO"
  иначе
    вывод minDigit
  все
кон
```

Паскаль

```
var N,digit,minDigit: longint;
begin
  readln(N);
  minDigit := N mod 10;
  while N > 0 do
  begin
    digit := N mod 10;
    if digit mod 2 = 0 then
      if digit < minDigit then
        minDigit := digit;
    N := N div 10;
  end;
  if minDigit = 0 then
    writeln('NO')
  else
    writeln(minDigit)
end.
```

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 трехзначного числа, при вводе которого приведенная программа, несмотря на ошибки, выдает верный ответ.

3. Найдите допущенные программистом ошибки и исправьте их. Исправление ошибки должно затрагивать только строку, в которой находится ошибка. Для каждой ошибки:

- выпишите строку, в которой сделана ошибка;
- укажите, как исправить ошибку, то есть 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.

Частые ошибки. Выпускники не понимают условие задания и не могут ответить на нужные вопросы. Также они часто выявляют только одну ошибочную строку вместо двух.

Частые ошибки еще связаны с тем, что школьники:

- используют одну переменную вместо другой;
- инициализируют переменную неверным значением;
- используют неправильные операции.

Как избежать. Учителю необходимо развивать у школьников умение анализировать условия задания. Также педагогу стоит обратить внимание учеников на то, что строки кода с синтаксическими ошибками не будут считаться правильным выбором ответа. Если выпускник укажет такие строки, то эксперты снизят ему баллы, даже если все решение задания будет считаться правильным.

Чтобы успешно выполнять задания подобного типа, выпускнику надо повторить:

- правила построения программы на языке программирования;
- правила работы с переменными;
- принцип работы алгоритма «ветвление»;
- способ записи ветвления на блок-схеме;
- принципы работы вложенных условных операторов.

Задание 25

Чтобы выполнить это задание, выпускнику надо написать короткую программу, которая состоит из 10–15 строк. Оптимальность решения и синтаксические ошибки не учитываются. В примере предлагается написать программу, которая меняет исходный массив и выводит его на экран.

Пример

Пример задания 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 измененный массив, каждый элемент выводится с новой строчки. 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

```
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
...

END
```

Python

```
# допускается также
# использовать две
# целочисленные переменные j и k
a = []
n = 30
for i in range(0, n):
    a.append(int(input()))
...
```

Алгоритмический язык

```
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
```

Паскаль

```
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.
```

C++

```
#include <iostream>
using namespace std;
const int N = 30;
int main() {
    long a[N];
    long i, j, k;
    for (i = 0; i < N; i++)
        cin >>  a[i];
    ...
    return 0;
}
```

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).

Частые ошибки. Типичные ошибки связаны с тем, что школьники не фиксируют изменение значений элементов массива, выходят за границы массива, ошибаются в организации цикла или не выводят ответ.

Как избежать. При подготовке выпускников учителю необходимо уделить внимание однопроходным алгоритмам. Они учитывают предложенные ограничения по переменным и просты в исполнении. Также учителю стоит напомнить выпускникам, что условие задачи написано на нескольких языках программирования, но решение можно писать на любом другом языке.

Чтобы успешно выполнять задания подобного типа, выпускнику надо повторить:

- что такое одномерные и двумерные массивы;
- как записать массив на языке программирования;
- какой цикл использовать для обработки массива.

Задание 26

Задание проверяет умение обосновать стратегию игры по заданному алгоритму. В задаче требуется выполнить три задания. Ответы на эти задания выступают шагами решения общей задачи или опираются на предыдущие пункты задачи и выводы.

Пример

Пример задания 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 четырех 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, то есть 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; $1 \le S \le 61$.

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, то есть не являющиеся выигрышными независимо от игры противника. Выполните следующие задания.

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, причем 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 ребрах 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.

Частые ошибки. Выпускники не имеют представления о выигрышной стратегии игры как наборе правил, по которым игрок отвечает на любой допустимый ход противника. Поэтому ученики указывают один или несколько вариантов развития игры, но не анализируют и не обосновывают эти варианты.

Как избежать. Учителю надо научить выпускников анализировать комбинаторные игры. При разработке заданий для подготовки педагогу нужно рассматривать похожие игры с разными формулировками. Также педагог должен рассказать выпускникам про выигрышную стратегию. Учителю стоит обратить внимание школьников на то, что для решения задач подобного типа применяется метод построения деревьев.

Задание 27

Задание проверяет умение выпускников составлять собственные программы, чтобы решать задачи средней сложности.

Пример

Пример задания 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 задается количество чисел n ($2 \le n \le 12\,000$). 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 приведенного 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.

Частые ошибки. Ученики часто допускают логические ошибки из-за того, что рассматривают не все возможные варианты расположения пар чисел в последовательности.

Как избежать. Особенность задания заключается в том, что к нему нельзя подготовиться заранее, потому что каждый год на экзамен выводится новая задача. Учителю вместе с учениками важно проработать два варианта решения. Первый вариант – простое переборное решение. Такое решение оценивается в два балла. Второй вариант – эффективное по времени и используемой памяти решение. Такое решение эксперты оценят в четыре балла.

Еще выпускников надо научить проверять условие задания и полученный результат. Это поможет не допускать логических ошибок в программе и не потерять баллы.