Повышение качества преподавания информатики в контексте анализа результатов мониторинговых исследований 2016 года и использования в образовательном процессе их результатов

На основе сравнительно-сопоставительного анализа результатов мониторинга оценки качества образовательных достижений обучающихся 10-х классов общеобразовательных организаций Республики Северная Осетия – Алания по углубленному изучению учебных предметов и (или) профильному обучению, а также на основе результатов ОГЭ по информатике, которые были получены в 2016 году выделены темы, которые были вынесены на итоговую аттестацию (ЕГЭ) в 2016 в соответствии с кодификатором, тем не менее обучающиеся показали низкие результаты по следующим трем основным разделам:

1. «Основы логики»;
2. «Элементы теории алгоритмов. Программирование»;
3. выполнение заданий с развернутым ответом.

Раздел информатики «Основы логики» основополагающий и тесно связан с другими разделами информатики, поэтому обеспечив должную подготовку по этому разделу, мы повысим успешность выполнения других разделов курса информатики. Этому разделу посвящено 4 задания КИМов ЕГЭ по информатике (№№2, 17 базового уровня, №18 повышенного и №23 высокого уровня сложности). Результативность выполнения №18 и №23 составляет в 2016 г. 8,5% и 3,5% соответственно.

Немаловажное значение имеет выбор УМК, следует отметить, что указанному разделу в УМК под редакцией Полякова К.Ю. в курсе информатики 10-11 классов профильного уровня отводится 10 часов, в УМК под редакцией Угринович Н.Д. и УМК под редакцией Семакина И.Г. по 18 часов, в то время как на базовом уровне в 10-11 классах на изучение этого раздела отводится не более 2-х часов.

Соответственно подготовка, которая может быть проведена по разделу «Основы логики» при изучении информатики на базовом уровне может быть осуществлена в рамках дополнительных консультативных мероприятий, через самообразование и через индивидуально-групповые занятия. Таким образом, серьезная подготовка по разделу «Основы логики» может быть осуществлена только на профильном уровне.

Задания №18 и №23 предполагают применение участниками экзамена знаний и умений в новых учебных ситуациях, т.е. подготовка к выполнению заданий такого типа не может быть осуществлена по шаблону. Но можно предложить действенный инструментарий, который позволит обучающемуся успешно выполнить любое по-новому сформулированное задание.

В соответствии с кодификатором задания раздела «Основы логики» проверяют набор тем и понятий: в ходе подготовки по данному разделу обучающимся необходимо знать

•формы мышления (понятия, суждения, умозаключения);

•основные логические функции;

•законы логики;

•методы решения логических уравнений и систем логических уравнений.

Задания из раздела «Основы логики» проверяют умения

•строить диаграммы Эйлера-Венна;

•строить таблицы истинности для сложных высказываний;

•строить и преобразовывать логические выражения;

•анализировать схемы, построенные с использованием базовых логических элементов компьютера и строить для них таблицы истинности.

Обучающийся должен

1. знать стандартный набор логических формул:
   a) свойства 0 и 1;
   b) свойство отрицания;
   c) основные законы логического сложения и логического умножения;
   d) свойства импликации и свойства эквивалентности.
2. Иметь твердое представление о понятии «множества», связи множеств и логических функций

Комплексная подготовка по разделу «Основы логики» заключается ее только в том, чтобы порешать определенный набор и тип заданий с обучающимися, а дать качественно теоретический материал и порешать с учениками сложные задачи этого раздела, тогда обучающиеся не будут «бояться» новых заданий КИМов и будут браться за их решение.

Рекомендуемая схема выполнения задания №18

1. **Преобразование логических выражений**
   - Определение элементарных высказываний.
   - Замена переменных (при необходимости).
   - Раскрытие импликации или эквиваленции.
   - Преобразование с использованием законов алгебры логики.
2. Построение таблицы истинности
3. Запись ответа

Рассмотрим задание 18 [1].

Обозначим через m&n поразрядную конъюнкцию неотрицательных целых чисел m и n. Так, например, 14&5 = 1110_2 & 0101_2 = 0100_2 = 4. Для какого наименьшего неотрицательного целого числа A формула x&25 ≠ 0 → (x&17 = 0 → x&A ≠ 0) тождественно истинна (т.е. принимает значение 1 при любом неотрицательном целом значении переменной x)?

Ответ: _____

Решение.

Введём обозначения: **P** = (x & 25 ≠ 0), **Q** = (x & 17 = 0), **A** = (x & A ≠ 0)

Перепишем исходное выражение и преобразуем его, используя свойство импликации:

$$(\mathbf{P} \to (\mathbf{Q} \to \mathbf{A}) = \mathbf{P} \to (\overline{\mathbf{Q}} + \mathbf{A}) = \overline{\mathbf{P}} + \overline{\mathbf{Q}} + \mathbf{A}$$

Формула $\overline{\mathbf{P}} + \overline{\mathbf{Q}} + \mathbf{A}$ =1, соответствует записи x&25 = 0 V (x&17 ≠ 0 V x&A ≠ 0).

Запишем двоичные представления всех заданных чисел:

$25 = 11001_2$

$17 = 10001_2$

Построим таблицу истинности для полученной формулы:

Число ***x***	Двоичный код числа ***x***	x&25 = 0	x&17 ≠0	x&A ≠ 0	
		$25 = 11001_2$	$17 = 10001_2$		
1	00001	0	1	любое	1
2	00010	1	0	любое	1
4	00100	1	0	любое	1

[1]Демонстрационный вариант КИМов для проведения в 2016 году ЕГЭ по ИНФОРМАТИКЕ и ИК

8	01000	0	0	1	1
16	10000	0	1	любое	1

Из таблицы истинности следует, что искомое число А должно содержать двоичный разряд на третьей позиции. Следовательно, искомое минимальное число равно 8.

Ответ: 8

Рассмотрим задание 18 [2].

Введём выражение M & K, обозначающее поразрядную конъюнкцию M и K (логическое «И» между соответствующими битами двоичной записи). Определите наименьшее натуральное число A, такое что выражение

$$((x \,\&\, 28 \neq 0) \vee (x \,\&\, 45 \neq 0)) \rightarrow ((x \,\&\, 48 = 0) \rightarrow (x \,\&\, A \neq 0))$$

тождественно истинно (то есть принимает значение 1 при любом натуральном значении переменной X)?

Решение:

Введём обозначения:

P = (x & 28 ≠ 0), **Q** = (x & 45 ≠ 0), **R** = (x & 48 ≠ 0), **A** = (x & A ≠ 0)

перепишем исходное выражение и преобразуем его, используя свойство импликации:

$$(\mathbf{P}+\mathbf{Q}) \rightarrow (\overline{\mathbf{R}} \rightarrow \mathbf{A}) = \overline{\mathbf{P}+\mathbf{Q}} + (\overline{\mathbf{R}} \rightarrow \mathbf{A}) = \overline{\mathbf{P}+\mathbf{Q}} + \mathbf{R} + \mathbf{A}$$

перейдем к импликации $(\overline{\overline{\mathbf{P}+\mathbf{Q}}+\mathbf{R}}) \rightarrow \mathbf{A} = ((\mathbf{P}+\mathbf{Q}) \cdot \overline{\mathbf{R}}) \rightarrow \mathbf{A}$.

Таким образом, для всех x, для которых $(\mathbf{P}+\mathbf{Q}) \cdot \overline{\mathbf{R}} = 1$, нам нужно обеспечить $\mathbf{A} = 1$

Условие $(\mathbf{P}+\mathbf{Q}) \cdot \overline{\mathbf{R}} = 1$ означает, что одновременно

а) истинно условие $\mathbf{P}+\mathbf{Q}$

б) истинно условие $\overline{\mathbf{R}}$

Запишем двоичные представления всех заданных чисел:

$28 = 11100_2$, $45 = 101101_2$, $48 = 110000_2$

P = 1 означает, что среди битов 4, 3 и 2 числа x есть единичные

Q = 1 означает, что среди битов 5, 3, 2 и 0 числа x есть единичные

Условие **P + Q = 1** означает, что выполняется **P** или **Q**

Условие $\overline{\mathbf{R}}$ говорит о том, что биты 5 и 4 числа x – нулевые

Объединяя все эти условия, находим, что «опасными» с точки зрения истинности заданного выражения могут быть два типа значений x:

а) числа, в которых могут быть ненулевыми биты 3 и 2 (при этом **P = 1**)

б) числа, в которых могут быть ненулевыми биты 3, 2 и 0 (при этом **Q = 1** и **P = 1**)

Очевидно, что тип б) включает в себя тип а).

Поэтому для того, чтобы все выражение было равно 1 при таких значениях x, нужно сделать единичными, по крайней мере, биты 3, 2 и 0 числа А.

Получается $A = 2^3 + 2^2 + 2^0 = 13$

Ответ: 13.

Задание 18.3[3]

На числовой прямой даны два отрезка: P = [30, 65] и Q = [10, 35]. Отрезок A таков, что

---

[2] *К. Поляков, 2009-2016*
http://kpolyakov.spb.ru/download/ege18.doc

[3] В.Р. Лещинер, М.А. Ройтберг Методические рекомендации для учителей, подготовленные на основе анализа типичных ошибок участников ЕГЭ 2016 года

формула

$$\neg(x \to A) \to ((x \to P) \to \neg (x \to Q))$$

истинна при любом значении переменной x.
Какова наименьшая возможная длина отрезка A?

Решение.

Для начала можно применить преобразование импликации два раза и получить выражение без импликации:

$\neg(x \in A) \to ((x \in P) \to \neg (x \in Q))$ равносильно выражению

$\neg(x \in A) \to ((x \notin P) \vee (x \notin Q))$

(мы преобразовали импликацию в скобках и применили отрицание, заменив «принадлежит» на «не принадлежит). Аналогично это выражение равносильно $(x \in A) \vee ((x \notin P) \vee (x \notin Q))$.

Скобки можно раскрыть, получаем $(x \in A) \vee (x \notin P) \vee (x \notin Q)$.

Для этого выражения формула истинна для всех x, не принадлежащих либо отрезку P, либо отрезку Q. Чтобы она была истинна для всей числовой прямой, требуется, чтобы отрезок A полностью покрывал пересечение отрезков P и Q. Минимальный такой отрезок [30, 35] совпадает с пересечением отрезков P и Q и имеет длину 5.
Ответ:5

Наибольшие затруднения у выпускников вызывает задание 23 ЕГЭ по информатике. Существует несколько методов решения этого задания:

Замена переменных
Последовательное подключение уравнений
Метод отображения
Метод с битовыми цепочками.

Рассмотрим «Метод отображения» для решения задания 23, представленном во фрагменте статьи «Люблю ЕГЭ за В15, или Еще раз про метод отображения»[4] Мирончик Е.А. Задание на решение системы логических уравнений остается в ЕГЭ одним из самых сложных. Но решение этой системы не только проверяет знание логических операций и умение считать у наших школьников, но и учит рассуждать, строить логические цепочки. Конечно, оно незаслуженно находится в части В. При оценке ответа нет возможности квалифицировать ошибку, так как ответ, как и логическое высказывание, бывает либо истинным, либо ложным. А поводов дать неверный в этом случае ответ много: можно написать наугад, а можно решить все от начала до конца, проделать все логические преобразования, выстроить верную цепочку рассуждений и в последнем действии допустить арифметическую ошибку. Заметим, что при решении этого задания количество только арифметических действий доходит до 40. Но тут у выпускников и учителей нет выбора. Будем действовать по схеме - сначала купили, потом полюбили. За что можно полюбить это задание? Например, за то, что оно не скучное, что в нем больше разнообразия, чем в задачах на количество информации, где надо просто применить формулу, или в задачах про системы счисления, в которых надо освоить три алгоритма. На примере задания В15 можно еще раз поговорить о сложных понятиях информатики: “деревья”, “графы”, “матрица смежности”. А самое главное, В15 учит думать и рассуждать.

Одним из ключевых моментов при разборе систем является определение основного

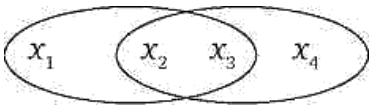
[4] Журнал «Информатика», 7-8, 2014

отображения, необходимого для решения системы. Обратите внимание, что в примерах знак «·» обозначает конъюнкцию, а «+» — дизъюнкцию.

Система 1.

$$\begin{cases} x_1 \cdot (x_2 + \overline{x_3}) + \overline{x_1} \cdot (x_2 \oplus x_3) = 1 \\ x_2 \cdot (x_3 + \overline{x_4}) + \overline{x_2} \cdot (x_3 \oplus x_4) = 1 \\ \dots \\ x_8 \cdot (x_9 + \overline{x_{10}}) + \overline{x_8} \cdot (x_9 \oplus x_{10}) = 1 \end{cases}$$

В этой системе в двух соседних уравнениях присутствует пара общих неизвестных



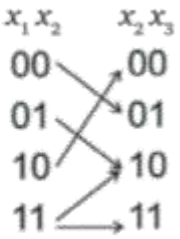
x5...

Зная количество пар (x1, x2), можно найти количество пар (x2, x3) и найти общее количество решений первого уравнения системы, а продолжая применять правило, построенное для первого уравнения, можно найти количество пар (x9, x10) и определить, сколько раз дерево решений доведет до x10, что и будет ответом к этому заданию. В цепочке рассуждений будем переходить от пары к паре:

$$(x1, x2) \Longrightarrow (x2, x3) \Longrightarrow (x3, x4) \Longrightarrow (x4, x5) \Longrightarrow \cdots \Longrightarrow (x9, x10)$$

Построим дерево решений первого уравнения и отображение, соответствующее первому уравнению:

x_1	x_2	x_3
0	0	1
	1	0
1	0	0
	1	0
		1



	Количество пар								
Пара	x_1, x_2	x_2, x_3	x_3, x_4	x_4, x_5	x_5, x_6	x_6, x_7	x_7, x_8	x_8, x_9	x_9, x_{10}
00	1	1	2	2	2	3	3	3	4
01	1	1	1	2	2	2	3	3	3
10	1	2	2	2	3	3	3	4	4
11	1	1	1	1	1	1	1	1	1

4+ 3 + 4 + 1 = 12
Ответ: 12

Система 2

$$\begin{cases} (x_1 \to x_2) \cdot (x_2 \to x_3) \cdot (x_3 \to x_4) \cdot (x_4 \to x_5) \cdot (x_5 \to x_6) = 1 \\ (x_1 \to y_1) \cdot (x_2 \to y_2) \cdot (x_3 \to y_3) \cdot (x_4 \to y_4) \cdot (x_5 \to y_5) \cdot (x_6 \to y_6) = 1 \end{cases}$$

Первый способ

Выражения в левой части первого и второго уравнений равны 1, следовательно, и произведение левых частей равно 1.

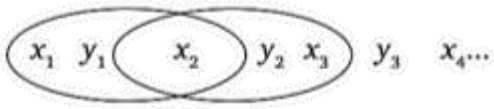
$$(x1 \to x2) \cdot \ldots \cdot (x5 \to x6) \cdot (x1 \to y1) \cdot \ldots \cdot (x5 \to x6) = 1$$

Перегруппируем множители:

$$(x1 \to x2) \cdot (x1 \to y1) \cdot \ldots \cdot (x5 \to x6) \cdot \ldots \cdot (x5 \to y5) \cdot (x6 \to x6) = 1$$

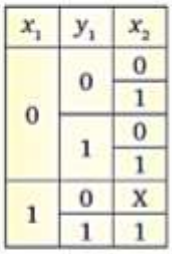
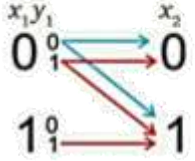
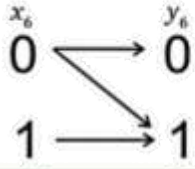
И запишем полученное уравнение в виде системы:

$$\begin{cases} (x_1 \to x_2) \cdot (x_1 \to y_1) = 1 \\ (x_2 \to x_3) \cdot (x_2 \to y_2) = 1 \\ (x_3 \to x_4) \cdot (x_3 \to y_3) = 1 \\ (x_4 \to x_5) \cdot (x_4 \to y_4) = 1 \\ (x_5 \to x_6) \cdot (x_5 \to y_5) = 1 \\ x_6 \to y_6 = 1 \end{cases}$$



Общими переменными для соседних уравнений является одна переменная.

x1 ⟹x2 ⟹x3 ⟹x4 ⟹x5 ⟹x6 ⟹y5

Для первого уравнения системы (и похожих на него 2, 3, 4, 5-го) дерево решений будет таким:	Первым пяти уравнениям будет соответствовать отображение:
	
Отображение для последнего уравнения:	

По построенному отображению заполним таблицу для вычисления количества решений:

	Количество решений по первым пяти уравнениям						Количество решений после подключения последнего уравнения
	x_1	x_2	x_3	x_4	x_5	x_6	y_6
0	1	2	4	8	16	32	32
1	1	3	7	15	31	63	95

Ответ: 32 + 95 = 127 решений

Второй способ

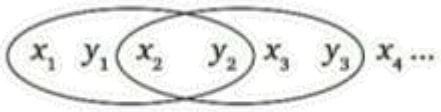
$$\begin{cases}(x_1 \to x_2)\cdot(x_2 \to x_3)\cdot(x_3 \to x_4)\cdot(x_4 \to x_5)\cdot(x_5 \to x_6)=1\\(x_1 \to y_1)\cdot(x_2 \to y_2)\cdot(x_3 \to y_3)\cdot(x_4 \to y_4)\cdot(x_5 \to y_5)\cdot(x_6 \to y_6)=1\end{cases}$$

Так же, как и в предыдущем случае, перепишем систему в виде:

$$\begin{cases}(x_1 \to x_2)\cdot(x_1 \to y_1)=1\\(x_2 \to x_3)\cdot(x_2 \to y_2)=1\\(x_3 \to x_4)\cdot(x_3 \to y_3)=1\\(x_4 \to x_5)\cdot(x_4 \to y_4)=1\\(x_5 \to x_6)\cdot(x_5 \to y_5)=1\\x_6 \to y_6=1\end{cases}$$

В первых пяти уравнениях по три переменных. Пара (x1, y1) определяет количество возможных сочетаний пары (x2, y2), а пара (x3, y3) получается из пары (x2, y2) по точно такому же правилу. Но в первом уравнении нет y2, а во втором не хватает y3 и так далее. Значит, первое уравнение не накладывает никаких ограничений на значения y2, а второе — на y3 и т.д. Значит, их значения могут быть любыми. Добавим недостающие переменные в первые пять уравнений, не изменяя систему.

$$\begin{cases}(x_1 \to x_2)\cdot(x_1 \to y_1)+y_2\cdot\overline{y_2}=1\\(x_2 \to x_3)\cdot(x_2 \to y_2)+y_3\cdot\overline{y_3}=1\\(x_3 \to x_4)\cdot(x_3 \to y_3)+y_4\cdot\overline{y_4}=1\\(x_4 \to x_5)\cdot(x_4 \to y_4)+y_5\cdot\overline{y_5}=1\\(x_5 \to x_6)\cdot(x_5 \to y_5)+y_6\cdot\overline{y_6}=1\\x_6 \to y_6=1\end{cases}$$

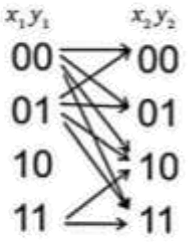


Каждое уравнение системы зависит от четырех переменных. Общими переменными для соседних уравнений является пара переменных.

(x1, y1) ⟹ (x2, y2) ⟹ (x3, y3) ⟹ (x4, y4) ⟹ (x5, y5) ⟹ (x6, y6)

Построим дерево решений первого уравнения и отображение множеств ему соответствующее.

x_1	y_1	x_2	y_2
0	0	0	0
			1
		1	0
			1
	1	0	0
			1
		1	0
			1
1	0	x	
	1	1	0
			1



При заполнении таблицы будем использовать это отображение пять раз и вычислим количество пар (x6, y6). Можно задать отображение матрицей смежности.

		приемник				Количество решений по первым пяти уравнениям					
		00	01	10	11	x_1,y_1	x_2,y_2	x_3,y_3	x_4,y_4	x_5,y_5	x_6,y_6
источник	00	+	+	+	+	1	2	4	8	16	32
	01	+	+	+	+	1	2	4	8	16	32
	10					1	3	7	15	31	63
	11			+	+	1	3	7	15	31	63

Последнему уравнению не удовлетворяет пара (10). Ответ будет складываться из значений последнего столбика, соответствующих парам (00), (01) и (11).
32 + 32 + 63 = 127
Ответ: 127

Раздел «Элемента теории алгоритмов. Программирование» в КИМых ЕГЭ по информатике представлен в трех заданиях - №№ 6, 8, 11 – базового уровня и в задания 14, 19, 20, 21, 22 повышенного уровня сложности.

Задание 6 требует от экзаменующихся формальное исполнение алгоритма, записанного на естественном языке.

Задание 8 подразумевает необходимость определить результат выполнения программы, записанной на языке программирования.

Задание 11 требует выполнение программы содержащей рекурсивный алгоритм. Следует отметь, что в 2016 году порядка 18% обучающихся справились с этим заданием.

Проверяет умение использовать алгоритм для конкретного исполнителя с фиксированным набором команд задание 14.

Задание на обработку массива – это задание 19.

В свою очередь задание 20 проверяет умение анализировать алгоритм, содержащий вспомогательный алгоритм либо циклы ил ветвления.

Сложным является 21 задание - на анализ программы с процедурами или функциями, уровень выполнения 18% по результатам 2016 года.

В зависимости от уровня изучения предмета колеблется количество часов отводимых в программах 10-11 классов по информатике на изучение раздела «Элементы теории алгоритмов. Программирование». Следует констатировать, что при базовом уровне изучения предмета освоить должным образом этот сложный раздел практически невозможно, т.к. на его изучение отводится не более 22 часов в 10-11 классах. В то время, как профильный уровень отводит на изучение раздела «Элементы теории алгоритмов. Программирование» от

81 (Семакин И.Г.) до 101 (Поляков К.Ю.) часа.

Для успешной подготовки к выполнению задания ЕГЭ по элементам теории алгоритмов и программированию обойтись урочными часами проблематично, поэтому необходимо предусмотреть систему дополнительных занятий.

В соответствии с кодификатором элементов содержания ЕГЭ выпускники должны иметь прочные знания о:

- порядке выполнения арифметических операций в программах, записанных на языках программирования;
- правилах выполнения линейных, разветвляющихся и циклических алгоритмов;
- операторах ветвления; цикла с параметром, предусловием и постусловием.

Выпускник должен умнеть:

- проводить анализ алгоритма построения последовательности;
- выполнять трассировку программы, записанной на языке программирования;
- написать программу, содержащую основные алгоритмические конструкции;
- оформлять программы содержащие подпрограммы, в том числе рекурсивные.

Соответственно экзаменующиеся должны в совершенстве владеть минимальным теоретическим материалом по разделу «Элементы теории алгоритмов. Программирование»:

- Алгоритм. Способы записи алгоритмов. Блок-схема.
- Правила выполнения линейных, разветвляющихся и циклических алгоритмов.
- Подпрограммы и их виды. Реализация на языках программирования.
- Рекурсивный алгоритм.
- Исполнитель алгоритма. Система команд исполнителя.
- Анализ алгоритма построения последовательности.
- Основы динамического программирования.
- Переменная, константа, операторы ввода/вывода, оператор присваивания.
- Полное и неполное ветвление, цикл с параметром, цикл с предусловием, цикл с постусловием.
- Простые и сложные условия в цикле и ветвлении.
- Массив: объявление, заполнение, вывод на экран, сортировка, отбор в соответствии с условиями. Обработка числовых и символьных массивов.
- Матрицы. Объявление и заполнение двумерных массивов. Операции над элементами двумерных массивов.
- Операции со строками. Основные операции с символьными строками. Обработка данных, вводимых в виде символьных строк.

Необходимо отметь, что с разделом «Элементы теории алгоритмов. Программирование» обучающиеся знакомятся уже в основной школе, а в 10-11 классах идет расширение и углубление представления об алгоритмизации и программировании.

Особенностью преподавания раздела «Элементы теории алгоритмов. Программирование» в школьном курсе, прежде всего, является четкая теоретическая база знаний по разделу с использованием трассировочных таблиц при объяснении теоретического материала.

Имея определенный набор знаний, следует приступить к совместному проектированию программы и только потом предложить обучающимся индивидуальную траекторию при организации самостоятельной работы, отличающейся уровнем сложности.

Для контроля знаний данного раздела обучающимся можно предложить дифференцированные задания разного уровня сложности, либо онлайн-тесты для подготовки к ОГЭ и ЕГЭ по информатике, размещенные на сайте К.Ю. Полякова http://kpolyakov.spb.ru/school/oge/online.htm.

Рекомендации по организации занятий при изучении раздела «Элементы теории алгоритмов. Программирование»: [5]

1. При изучении алгоритмических структур давать задания на построение и анализ
2. Использование практикумов для системы Кумир с автоматической проверкой выполненных заданий

Источники: Программа Кумир версии 2.1. и выше

- Сайт К.Ю. Полякова: http://kpolyakov.spb.ru\kumir.html
- Сайт Д. Кириенко http://server179.ru

3. Использование сервисов с автоматической проверкой программ.

Ресурсы:

- http://informatics.mccme.ru/
- http://www.acmp.ru

4. Изучение различных видов формальных исполнителей в курсе информатики 5-10-ых классов
5. Изучение стандартных алгоритмов обработки данных.
6. Использование дополнительных учебных пособий по программированию.
7. Популяризация программирования как профессиональной деятельности.
   - Разработка программ внеурочной деятельности и курсов по выбору.
   - Проведение внеклассных мероприятий;
   - Участие в различных соревнованиях и олимпиадах.

В базовом курсе информатики рекурсивным алгоритмам (задание 11) и подпрограммам (задание 21) не уделяется должного внимания из-за дефицита времени.

Задание 11 проверяет владение экзаменуемыми понятием рекурсии в алгоритмах и связанных с этим понятием умений и навыков. Существует решение этого задания методом формального исполнения (трассировки) алгоритма, то есть в результате репродуктивной деятельности, знакомой учащимся, хотя более простым для реализации является решение методом записи рекуррентных соотношений и построения таблицы значений. Низкий показатель выполнения этого задания говорит о том, что понятие рекурсии многими обучающимися в процессе обучения так и не было освоено.

При выполнении заданий **с** рекурсивным алгоритмом **нужно знать**:

- рекурсия – это приём, позволяющий свести исходную задачу к одной или нескольким более простым задачам того же типа;
- чтобы определить рекурсию, нужно задать:
  1. условие остановки рекурсии (базовый случай или несколько базовых случаев)
  2. рекуррентную формулу
- любую рекурсивную процедуру можно запрограммировать с помощью цикла;
- рекурсия позволяет заменить цикл и в некоторых сложных задачах делает решение более понятным, хотя часто менее эффективным.

Задание 11.1 [6]

---

[5] Филиппов В. И. Методика подготовки учащихся к выполнению заданий из разделов «Элементы теории алгоритмов» и «Программирование» контрольно-измерительных материалов ЕГЭ по информатике и ИКТ

[6] В.Р. Лещинер, М.А. Ройтберг Методические рекомендации для учителей, подготовленные на основе анализа типичных ошибок участников ЕГЭ 2016 года

Ниже на пяти языках программирования записана рекурсивная функция (процедура) F.

Бейсик

```
SUB F(n)
  print n,
  IF n >= 7 THEN
    F(n - 3)
    F(n - 1)
  END IF
END SUB
```

Python

```
def F(n):
    print(n, end='')
    if n >= 7:
        F(n - 3)
        F(n - 1)
```

Алгоритмический язык

```
алг F(цел n)
нач
  вывод n
  если n >= 7 то
    F(n - 3)
    F(n - 1)
  все
кон
```

Паскаль

```
procedure F(n: integer);
begin
  write(n);
  if n >= 7 then
  begin
    F(n - 3);
    F(n - 1)
  end
end;
```

Си

```
void F(int n) {
  printf("%d", n);
  if (n >= 7) {
    F(n - 3);
    F(n - 1);
  }
}
```

Что выведет программа при вызове F(9)? В ответе запишите последовательность выведенных цифр слитно (без пробелов).

Решение

Сначала необходимо изучить текст программы на одном из языков программирования и понять, что выполняет данная функция. Функция получает на вход одно число n, выводит его на экран, затем при условии, что $n \geq 7$ осуществляет два последовательных вызова F(n − 3) и F(n − 1), что приведет к печати меньших значений n и дальнейшим рекурсивным вызовам.

Например, при данном $n = 9$ программа напечатает число 9, затем вызовет F(6), то есть после числа 9 будет напечатано то, что выведет функция при вызове F(6), затем произойдет вызов F(8). Упрощенно это можно записать так: F(9) = 9, F(6), F(8), то есть ответ будет представлять собой последовательную запись (конкатенацию) цифры 9, ответа для F(6) и ответа для F(8).

Выпишем рекуррентное соотношение для общего случая:

F(n) = n, F(n − 3), F(n − 1), при $n \geq 7$;

F(n) = n, при $n < 7$.

Далее заполним таблицу, что выведет функция при вызове для разных значений n:

n	Рекуррентное соотношение для F(n)	Результат вызова функции F(n)
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5

6	6	6
7	7, F(4), F(6)	746
8	8, F(5), F(7)	85746
9	9, F(6), F(8)	9685746

Например, вызов F(7) приведет к печати цифр 746, так как вызовы F(4) и F(6) только напечатают цифры 4 и 6 и не будут совершать никаких дальнейших рекуррентных вызовов. Вызов F(8) напечатает 8, затем сделает вызов F(5), затем сделает вызов F(7). Вызов F(5) напечатает одну цифру 5, а вызов F(7), как было определено ранее, напечатает 746, поэтому ответом для F(8) будет 85746.

Наконец, F(9) напечатает 9, сделает вызов F(6), который напечатает 6, и сделает вызов F(8), который напечатает 85746. Последовательно записав эти цифры, получим ответ для **F(9): 9685746**

Задание Р-05[7]. Дан рекурсивный алгоритм:

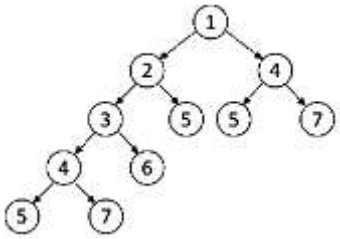
```
procedure F(n: integer);
    begin
        writeln(n);
        if n < 5 then begin
            F(n + 1);
            F(n + 3)
        end
    end;
```

Найдите сумму чисел, которые будут выведены при вызове F(1).

Решение (вариант 1, построение дерева вызовов):

1) поскольку в начале каждого вызова на экран выводится значение единственного параметра функции, достаточно определить порядок рекурсивных вызовов и сложить значения параметров;

2) поскольку при n<5 выполняется два рекурсивных вызова, решать такую задачу «на бумажке» удобно в виде двоичного дерева (в узлах записаны значения параметров при вызове функции):



Складывая **все** эти числа, получаем 49)

Ответ: 49.

Решение (вариант 2, подстановка):

1) можно обойтись и без дерева, учитывая, что при каждом вызове с n < 5 происходит два рекурсивных вызова; сумму чисел, полученных при вызове $F(n)$, обозначим через $S(n)$:

[7]К. Ю. Поляков, http://kpolyakov.spb.ru/download/ege11.doc

$$S(n) = \begin{cases} n + S(n+1) + S(n+3), & n < 5 \\ n, & n \geq 5 \end{cases}$$

2) выполняем вычисления:

$S(1) = 1 + S(2) + S(4)$

$S(2) = 2 + S(3) + S(5) = 7 + S(3)$

$S(3) = 3 + S(4) + S(6) = 9 + S(4)$

$S(4) = 4 + S(5) + S(7) = 16$

3) теперь остаётся вычислить ответ «обратным ходом»:

$S(3) = 9 + 16 = 25$

4) $S(2) = 7 + 25 = 32$

$S(1) = 1 + 32 + 16 = 49$

5) Ответ: 49.

Эффективным способом организации занятий по данной теме может быть разбор заданий у доски с объяснением учителя, а затем самостоятельная работа учащихся с теми же алгоритмами, реализованными в среде программирования (например, в среде «КуМир»), когда учащиеся могут запустить эти алгоритмы с разными аргументами и обсудить, в чем причина совпадения или несовпадения результатов. Далее учащиеся могут попробовать модифицировать алгоритмы (например, переместить операторы вывода) и спрогнозировать результаты их выполнения, а затем проверить гипотезу.

В 2016 году 19,8% участников ЕГЭ по информатике приступили к выполнению заданий с развернутым ответом.

Для успешного выполнения и получения максимальных трех баллов за решение задания 24 обучающиеся должны иметь твердое представление об арифметических операциях и порядок их выполнения в программах, записанных на языках программирования; знать структуры ветвления, циклы различных видов и их реализация на языках программирования; иметь выполнять трассировку программы, записанной на языке программирования.

Задание 24 проверяет умение обучающихся прочитать фрагмент программы на языке программирования и исправить допущенные ошибки. В данном задании необходимо указать результаты работы программы при введении определённой последовательности входных данных; привести пример входных данных, при которых программа работает верно; найти и указать две ошибки (две ошибочных строки программы); исправить обе ошибки.

При выполнении первого действия ошибки связаны с указанием неверного значения; в то время как при выполнении второго действия часто приводят пример неверного числа.

При выполнении третьего и четвертого действий экзаменующиеся не выписывают строку с ошибкой, а указывают её номер или приводят программу целиком. Часто, учащиеся находят и исправляют только одну ошибку.

Ниже представлен вариант решения задания 24 Поляковым К.Ю[8].:

На обработку поступает последовательность из четырёх неотрицательных целых чисел (некоторые числа могут быть одинаковыми). Нужно написать программу, которая выводит на экран количество нечётных чисел в исходной последовательности и максимальное нечётное число. Если нечётных чисел нет, требуется на экран

[8] http://kpolyakov.spb.ru/download/ege24-C1.doc

вывести «NO». Известно, что вводимые числа не превышают 1000. Программист написал программу неправильно. Вот она:

```
const n = 4;
var i, x: integer;
var maximum, count: integer;
begin
  count := 0;
  maximum := 999;
  for i := 1 to n do begin
    read(x);
    if x mod 2 <> 0 then begin
      count := count + 1;
      if x > maximum then maximum := i
    end
  end;
  if count > 0 then begin
    writeln(count);
    writeln(maximum)
  end
  else writeln('NO')
end.
```

Последовательно выполните следующее.

1. Напишите, что выведет эта программа при вводе последовательности: 2 9 4 3

2. Приведите пример такой последовательности, содержащей хотя бы одно нечётное число, что, несмотря на ошибки, программа печатает правильный ответ.

3. Найдите все ошибки в этой программе (их может быть одна или несколько). Известно, что каждая ошибка затрагивает только одну строку и может быть исправлена без изменения других строк. Для каждой ошибки:

1) выпишите строку, в которой сделана ошибка;

2) укажите, как исправить ошибку, т.е приведите правильный вариант строки.

Обратите внимание, что требуется найти ошибки в имеющейся программе, а не написать свою, возможно, использующую другой алгоритм решения. Исправление ошибки должно затрагивать только строку, в которой находится ошибка.

Решение:

1) обратим внимание на две строки в начале программы, которые начинаются с ключевого слова **var**: **это не ошибка**, такое повторение, действительно, допустимо в языке Паскаль; возможно, это была одна из ловушек разработчиков КИМ, которую они применили на реальном ЕГЭ-2014
2) теперь выполним программу для заданной последовательности 2 9 4 3, записывая все изменения переменных в таблицу:

	условие	i	x	maximum	count	вывод
count:=0;					0	
maximum:=999;				999		
for i := 1 to n do		1				
read(x);			2			
if x mod 2 <> 0 then	нет					
end		2				

read(x);			**9**			
if x mod 2 <> 0 then	**да**					
count:=count+1;					**1**	
if x > maximum then	**нет**					
end		**3**				
read(x);			**4**			
if x mod 2 <> 0 then	**нет**					
end		**4**				
read(x);			**3**			
if x mod 2 <> 0 then	**да**					
count:=count+1;					**2**	
if x > maximum then	**нет**					
end		**5**				
if count > 0 then	**да**					
writeln(count);						**2**
writeln(maximum)						**999**

3) при ручной прокрутке программы мы увидели, что она правильно подсчитала количество нечётных чисел во входной последовательности, но неверно определила максимум: значение переменной **maximum**, которое было выведено на экран, осталось равным начальному значению 999, так как все остальные нечётные числа были меньше этого начального значения; поэтому ответ на п. 1 задания должен быть таким:

1) Программа выведет числа 2 и 999.

4) поскольку все числа по условию неотрицательны и не превышают 1000, программа всегда будет выдавать 999 вместо максимального нечётного числа; в то же время мы выяснили, что количество нечётных чисел в последовательности считается правильно; поэтому любая последовательность, содержащая 999, будет обрабатываться правильно
5) таким образом, правильный ответ на п. 2 должен быть таким:

2) Программа работает правильно для последовательности: 2 9 3 999.

6) теперь будем искать ошибки; как уже отмечалось, повторное использование ключевого слова **var** допустимо и указывать это в качестве ошибки нельзя!
7) как следует из результатов ручной прокрутки программы, во многих случаях она выдаёт неверный результат из-за того, что неверно задано начальное значение переменной **maximum**: оно должно быть **меньше**, чем любой возможный результат;
8) наименьшее нечётное неотрицательное число – это 1, поэтому можно принять в качестве начального значения **maximum** любое число, меньшее единицы (на самом деле, программа будет правильно работать и для 1), например:

3) **Ошибка 1**. **maximum:=999;**
Исправление: **maximum:=0;**

9) если теперь (с исправленной первой ошибкой) сделать ручную прокрутку программы, то мы увидим, что на последовательности 2 9 4 3 она выдает сначала 2, а потом – 4, то есть, значение максимума вычисляется опять неверно;
10) откуда появится число 4 в переменной **maximum**? оно будет записана в результате выполнения оператора **if x > maximum then maximum:=i**, который записывает в

переменную **maximum** не значение полученного числа (**x**), а его номер (**i**); таким образом, мы нашли вторую ошибку:

Ошибка 2. **if x > maximum then maximum:=i**
Исправление: **if x > maximum then maximum:=x**

Для того чтобы повысить качество знаний обучающихся при подготовке к ЕГЭ по информатике СОРИПКРО рекомендует учителям:

- добиваться глубокого понимания обучающимися каждой темы, каждого раздела учебного предмета;
- дать обучающимся достаточную практику применения полученных знаний и освоенных умений при решении заданий разных типов и моделей;
- широко использовать тематические сборники заданий в формате ЕГЭ,
- совместно с обучающимся разработать индивидуальную траекторию ликвидации пробелов в подготовке сдачи ЕГЭ;
- использовать дополнительную, послеурочную или элективную форму подготовки выпускников;
- для контроля знаний, умений и навыков обучающихся использовать для этого задания, аналогичные заданиям экзаменационных материалов;
- предложить список учебных пособий и ресурсов для самостоятельной подготовки к экзамену.