Методические указания по Отраслевой олимпиаде школьников «Газпром», профиль «Информационные и коммуникационные технологии»

Учебное пособие для подготовки к олимпиаде

Под редакцией Ю.Б.Буркатовской, В.В.Видмана

Тематика заданий

1. Системы счисления.
2. Обработка данных, программирование.
3. Логика.
4. Логическая цепочка.
5. Графы.
6. Шифрование.
7. Системы счисления.
8. Булева алгебра.
9. Кодирование и декодирование.
10. Количество информации.

Задание 1. Системы счисления

Задача 1. Джон и Брон работают на предприятии Сибирская Нефть. Долгие годы они работали в одной смене, и их любимым занятием были настольные игры. Но затем их распределили в разные смены, но и тут они нашли способ некоторого общения и развлечения: они оставляют друг другу разного рода загадки и ребусы. В этот раз Джон установил на шкаф с запасами кофе кодовый замок. Если Брон не сумеет отгадать код, то точно избавится от привычки пить кофе по утрам. Джон оставил записку со следующим текстом:

«В качестве ответа введи инверсию от полученного двоичного числа» и пример [BD(16) + 1323(4)] / 1100(2)

Представить ответ в виде двоичного числа.

Примечание: Числа в скобках () – обозначают систему счисления, в которой записано число.

Решение задачи 1.

Сначала необходимо привести все числа в одну систему исчисления. Это может быть любая удобная для вас система, в том числе можно переводить сразу в двоичную. Мы для примера возьмем 10-ю систему счисления:

BD(16) = 189(10)

1323(4) = 123(10)

1100(2) = 12(10)

В результате мы можем провести математические операции и получить в качестве ответа одно число в 10-й системе:

(189+123)/12 = 26

Следующим шагом мы переведем это число в 2-ю систему счисления. Над полученным числом нужно проделать инверсию, т.е. нули заменить единицами, а единицы заменить нулями.

26(10) = 11010(2)

11010 ---> 00101

Ответ: 00101.

Задачи в других вариантах аналогичны, поэтому приводится решение только для одного варианта. Числа можно переводить в любую систему счисления, соблюдая правила перевода.

Задание 2. Обработка данных и программирование

Задача 2. Давление газа, подаваемого из скважины, измеряется каждые 15 минут, в результате получается последовательность значений – вещественных чисел в диапазоне от 0 до 20. *Спадом давления* будем называть последовательность чисел, в которой предыдущее число больше последующего. *Величиной спада* будем называть частное от деления первого элемента спада на последний элемент. При нормальной работе скважины спад включает не более чем k значений, и величина спада не превышает m. Если хотя бы одно из условий не выполняется, это считается опасной ситуацией. Требуется разработать алгоритм подсчета количества опасных ситуаций на всем участке измерений и написать программу, реализующую этот алгоритм. Известно, что измерения производятся не более суток. Гарантируется, что имеется хотя бы один спад.

Входные данные. В первой строке содержится целое число n - количество измерений. Во второй строке содержится целое число k, в третьей – вещественное число m. В следующих строках содержится n вещественных чисел – результаты измерений.

Выходные данные. Целое число – количество опасных ситуаций на всем участке измерений.

Пример входных данных.

15

3

2

8.24

9.12

3.93

8.99

6.40

5.47

3.51

4.44

9.35

6.75

5.15

9.46

10.82

7.65

8.64

Пример выходных данных (для данного примера входных данных)

2

Пояснение. В последовательности 4 спада (выделены разными цветами).

Первый элемент	Последний элемент	Кол-во элементов	Величина
9,12	3,93	2	2,32
8,99	3,51	4	2,56
9,35	5,15	3	1,81
10,82	7,65	2	1,41

Опасная ситуация наблюдается в первом и втором спаде (выделено красным).

Решение задачи 2.

Данные о давлении могут быть сохранены в массиве (списке для Python) $A[]$ вещественных чисел. Поскольку измерения проводятся не более суток каждые 15 минут, максимальный размер массива будет 24×4=96. Рассматривается заданное число наблюдений $n \leq 96$. Чтобы выделить спад давления, нужно найти такую последовательность элементов массива $A[j] \ldots A[j+p]$, в которой $A[i] > A[i+1]$ для всех $i = j, \ldots, j+p-1$, и в эту последовательность нельзя добавить ни одного элемента так, чтобы сохранялось данное свойство. Количество элементов в спаде при этом равно $p+1$, а величина спада – $A[j]/A[j+p]$. Опасная ситуация возникает в случае выполнения **одного** из условий: $p+1 > k$ или $A[j]/A[j+p] > m$. Итак, находим очередной спад и подсчитываем его характеристики, если условие опасной ситуации выполняются, увеличиваем счетчик опасных ситуаций на 1 (в начала программы обнуляем счетчик).

В задачах подобного рода требуется еще отслеживать выход за границу массива и, возможно, отдельно обрабатывать спад, если он находится в конце массива. В решениях участников достаточно часто встречалась ошибка, связанная с обработкой последнего спада. Чтобы упростить эту задачу, можно использовать метод **барьерного элемента**: искусственно добавить в конец массива элемент, больше последнего. Тогда признаком конца спада всегда является условие $A[i] \leq A[i+1]$.

Ниже приведена программа на языке Pascal.

```
Program Problem_2;
var     i,n,k,count,beg,len: integer;
   val,m: real;
   A: array[1..97] of real;
begin
   readln(n);
   readln(k);
   readln(m);
   for i:=1 to n do
        readln (A[i]);
```

```
   A[n+1]:=A[n]+1;
   count:=0;
   beg:=1;
   i:=1;
   while(i<=n) do
   begin
       while(A[i]>A[i+1]) do
           i:=i+1;
       len:=i-beg+1;
       val:=A[beg]/A[i];
       if (len>k) or (val>m) then
           count:=count+1;
       beg:=i+1;
       i:=i+1;
   end;
   writeln(count);
end.
```

Ниже приведена программа на языке C++.

```
#include<iostream>
using namespace std;
int main()
{
int i, n, k, count, beg, len;
   float val,m, A[97];
   cin >> n >> k >> m;
   for(i=0;i<n;i++)
        cin >> A[i];
   A[n]=A[n-1]+1;
   count=0;
   beg=0;
   i=0;
    while(i<n)
    {
        while(A[i]>A[i+1]) i++;
```

```
        len=i-beg+1;
        val=A[beg]/A[i];
        if((len>k)||(val>m)) count++;
        i++;
        beg=i;
    }
  cout << count;
return 0;
}
```

Ниже приведена программа на языке Python.

```
n=int(input())
k=int(input())
m=float(input())
A=[]
for i in range(n):
    A.append(float(input()))
A.append(A[n-1]+1)
count=0;
i=0;
beg=0;
while i<n:
  while A[i]>A[i+1]:
       i+=1
  len=i-beg+1
  val=A[beg]/A[i]
  if len>k or val>m:
      count+=1
  i+=1
  beg=i
print(count)
```

Заметим, что приведенная программа не является единственным вариантом, возможны и другие решения. В частности, можно обойтись вообще без массива, если считывать очередное значение $a2$ и сравнивать его с предыдущим $a1$. При этом признаком конца спада

может служить как выполнение неравенства $a2 \geq a1$, так и $a2 < a1$ и $i = n$, где i – номер текущего наблюдения.

Ниже приведена программа на языке Pascal.

```
Program Problem_2;
var
i,n,k,count,beg,len: integer;
   val,m,a1,a2,abeg: real;
begin
   readln(n);
   readln(k);
   readln(m);
   readln(a1);
   count:=0;
   beg:=1;
   abeg:=a1;
   for i:=2 to n do
   begin
        readln (a2);
        if(a2>=a1) then begin
           len:=i-beg;
           val:=abeg/a1;
           if (len>k) or (val>m) then
               count:=count+1;
           beg:=i;
           abeg:=a2;
       end
       else if (i=n) then begin
           len:=i-beg+1;
           val:=abeg/a2;
           if (len>k) or (val>m) then
               count:=count+1;
       end;
       a1:=a2;
   end;
```

```
   writeln(count);
end.
```

Ниже приведена программа на языке C++.

```
#include<iostream>
using namespace std;
int main()
{
int i, n, k, count, beg, len;
   float val,m, a1,a2,abeg,aend;
   cin >> n >> k >> m >> a1;
   count=0;
   abeg=a1;
   beg=0;
   for(i=1;i<n;i++)
   {
        cin >> a2;
        if(a1<=a2)
        {
            len=i-beg;
            val=abeg/a1;
            if((len>k)||(val>m)) count++;
            beg=i;
            abeg=a2;
        }
        else if (i==n-1)
        {
            len=i-beg+1;
            val=abeg/a2;
            if((len>k)||(val>m)) count++;
        }
        a1=a2;
   }
   cout << count;
return 0;
```

Ниже приведена программа на языке Python.

```
n=int(input())
k=int(input())
m=float(input())
a1=float(input())
count=0;
i=0;
beg=0;
abeg=a1
for i in range(1,n):
    a2=float(input())
    if a1<=a2:
        len=i-beg
        val=abeg/a1
        if len>k or val>m:
            count+=1
        abeg=a2
        beg=i
    elif i==n-1:
        len=i-beg+1
        val=abeg/a2
        if len>k or val>m:
            count+=1
    a1=a2
print(count)
```

В других вариантах задача 2 имеет другие условия, но по сложности и по идее решения все задачи близки, поэтому мы приводим решение только для одного варианта.

Задание 3. Логика

Задача 3. Геологи Джон, Тирион, Иннокентий и Серсея работали в одной партии на месторождении Дотракия. Спустя некоторое время оказалось, что отчет об обработке важных образцов потерян. Геологи пытаются вспомнить, кто обрабатывал данные образцы. Вот их воспоминания.

Джон. Над образцами работали я или Тирион. Серсея не прикасалась к этим образцам.

Тирион. Иннокентий точно не работал над этими образцами вместе с Серсеей. Думаю, их обрабатывали Джон или Иннокентий.

Иннокентий. Мы с Серсеей либо оба работали с этими образцами, либо оба не работали. Во всяком случае, Тирион и я не работали с образцами вместе.

Серсея. Я не работала с образцами. Думаю, Джон или Тирион работали.

Начальник геологов, Семен Семеныч, помнит, что с образцами работали двое. Воспоминания геологов настолько противоречивы, что он предположил, что каждый из них ошибается в одном утверждении, и не ошибается в другом. Оказалось, что он был прав, и это дало возможность восстановить истину.

Кто же из геологов работал с образцами?

Решение задачи 3.

Переведем высказывания геологов с естественного языка на язык алгебры логики. Пусть символы D, T, I, S означают соответственно, что Джон, Тирион, Иннокентий и Серсея работали с образцами.

Имя	Первое высказывание	Второе высказывание
Джон	$D \vee T$	$\bar{S}$
Тирион	$\bar{I}\bar{S}$	$D \vee I$
Иннокентий	$I \equiv S$	$\overline{IT}$
Серсея	$\bar{S}$	$D \vee T$

Далее задачу можно решить методом гипотез. Предположим, Серсея работала с образцами, то есть $S = 1$ ($\bar{S} = 0$). Выпишем высказывания, помним, что одно из высказываний у каждого геолога принимает значение 1, второе – значение 0. Поэтому мы сразу можем определиться с высказываниями Серсеи и Иннокентия.

Гипотеза	Джон		Тирион		Иннокентий		Серсея	
	$D \vee T$	$\bar{S}$	$\bar{I}\bar{S}$	$D \vee I$	$I \equiv S$	$\overline{IT}$	$\bar{S}$	$D \vee T$

$S=1$	1	0					0	1

С образцами работали двое, первая – Серсея, второй – Джон или Тирион. Значит, Иннокентий точно не работал с образцами ($I=0$). Это помогает определиться с высказываниями Тириона и Иннокентия: $\overline{IS}=1$, $I \equiv S=0$, $\overline{IT}=1$. Второе высказывание Тириона принимает противоположное значение: $D \vee I=0$.

Гипотеза	Джон		Тирион		Иннокентий		Серсея	
	$D \vee T$	$\bar{S}$	$\overline{IS}$	$D \vee I$	$I \equiv S$	$\overline{IT}$	$\bar{S}$	$D \vee T$
$S=1, I=0$	1	0	1	0	0	1	0	1

Из того, что $D \vee I=0$ теперь следует $D=0$. Тогда $T=1$. Итак, над образцами работали Тирион и Серсея.

Гипотеза	Джон		Тирион		Иннокентий		Серсея	
	$D \vee T$	$\bar{S}$	$\overline{IS}$	$D \vee I$	$I \equiv S$	$\overline{IT}$	$\bar{S}$	$D \vee T$
$S=1$ $T=1$ $D=0$ $I=0$	1	0	1	0	0	1	0	1

Если бы в ходе рассуждений мы приходим к противоречию, следует рассмотреть противоположную гипотезу (Серсея не работала с образцами).

Предположим, Серсея не работала с образцами, то есть $S=0$ ($\bar{S}=1$). Выпишем высказывания, помним, что одно из высказываний у каждого геолога принимает значение 1, второе – значение 0. Поэтому мы сразу можем определиться с высказываниями Серсеи и Иннокентия.

Гипотеза	Джон		Тирион		Иннокентий		Серсея	
	$D \vee T$	$\bar{S}$	$\overline{IS}$	$D \vee I$	$I \equiv S$	$\overline{IT}$	$\bar{S}$	$D \vee T$
$S=0$	0	1					1	0

Получается, что ни Джон, ни Тирион не работали с образцами. Но этого не может быть, потому что с образцами работали двое. Значит, данная гипотеза неверна.

Следует отметить, что задача может допускать более одного решения; достаточно было найти одно.

Ответ. Тирион и Серсея.

Задание 4. Логическая цепочка

Задача 4. В целях безопасности на вашем компьютере каждый день меняется пароль. Четыре дня назад пароль был 2. Три дня назад паролем являлось число 10, два дня назад это было 42. Вчера пароль был уже 170. Какой же пароль сегодня? Между всеми числами определенно есть какая-то связь.

Ответом является число.

Решение задачи 4.

Это задача на логику и нужно найти связь между числами. Выпишем все числа в ряд.

2 10 42 170

В данном случае если перевести все числа в двоичную систему исчисления, то получится следующий ряд чисел

10 1010 101010 10101010

то следующим будет с пятью подряд 10, а это 682 в 10-й системе счисления

Ответ: 682

Задание 5. Графы

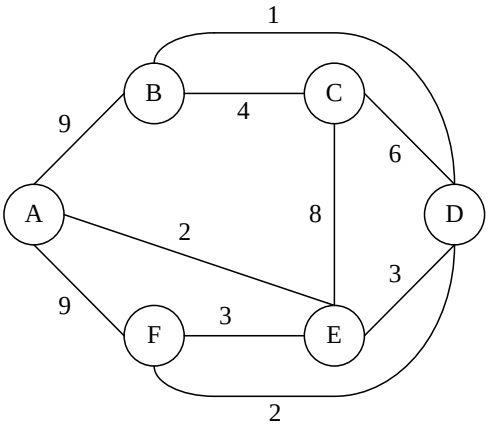
Задача 5. На месторождении шесть пунктов, в которых расположены скважины для добычи газа. Ежедневно руководство должно посещать все скважины. Сейчас решается вопрос, на какой скважине лучше устроить базу руководства, и в каком порядке объезжать скважины, чтобы тратить на дорогу как можно меньше времени.

В таблице указаны длины дорог между пунктами, пустая клетка означает, что дороги не существует (кроме клеток на главной диагонали). В ответе нужно указать, где расположить базу руководства, точный маршрут и его длину.

	A	B	C	D	E	F
A		9			2	9
B	9		4	1		
C		4		6	8	
D		1	6		3	2
E	2		8	3		3
F	9			2	3	

Решение задачи 5

Приведенную информацию о скважинах и дорогах можно представить в виде диаграммы графа. Вершинам соответствуют скважины, ребрам – дороги, числа возле ребер обозначают длины дорог. Если вы не знакомы с математическим понятием «граф», можно начать с Википедии, либо любого другого источника. В данном случае мы имеем дело с *простым взвешенным графом*: ребра не имеют направления, вершины соединены не более чем одним ребром, ребрам приписаны числа, называемые *весами*.



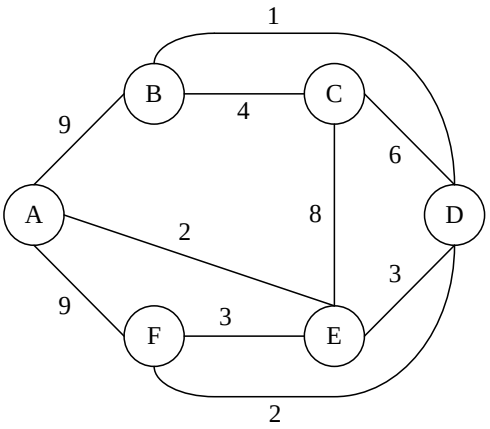
Вершины в графе называются *смежными*, если они соединены ребром.

Маршрутом в простом графе называется последовательность вершин $v_0v_1 \dots v_iv_{i+1} \dots v_{n-1}v_n$, в которой любые две соседние вершины смежны. Если $v_0 = v_n$, маршрут называется *замкнутым*, иначе – *открытым*.

Итак, задача состоит в построении замкнутого маршрута с минимальной суммой весов ребер. Это модификация *задачи коммивояжера*, в которой предполагается, что ребро есть между любой парой вершин, а каждая вершина посещается ровно один раз.

Не существует эффективного алгоритма решения подобных задач, приходится либо перебирать все возможные варианты (или отбрасывать часть заведомо плохих вариантов), или решать задачу приближенно, то есть находить не оптимальное решение, а близкое к нему.

Рассмотрим данный граф.



Видно, что в вершину A лучше всего попадать из вершины E, перемещаться же из A в вершины B и F невыгодно, лучше, например, вернуться в E и оттуда уже переместиться в F, затем в D. Из D уже можно посетить B и C. Стартовали мы из вершины E, можно теперь из C вернуться туда по ребру CE, либо по пути $CBDE$, вес в данном случае один и тот же. Итак, получили маршрут $EAEFDBCE$, общий вес $2 + 2 + 3 + 2 + 1 + 4 + 8 = 22$. Перебор всех вариантов тут является достаточно трудоемким, но, возможно, вам удастся найти другое решение. Заметим, что маршрут замкнутый, можно начинать с любой вершины.

Ответ. Маршрут $EAEFDBCE$, общий вес 22.

Задачи в других вариантах аналогичны, поэтому приводится решение только для одного варианта.

Задание 6. Шифрование

Задача 6. Информация, поступающая с месторождения, рассматривается как набор байтов. Для защиты от промышленного шпионажа каждый байт шифруется с использованием следующих операций:

1) Циклический сдвиг влево на одну позицию (например, байт 10011100 преобразуется в байт 00111001);

2) Инверсия (например, инверсия байта 10011100 есть 01100011);

3) Побитовое сложение по модулю два с маской (например, если байт равен 10011100, а маска – 11010010, то результат их сложения 01001110).

Каждый байт задает целое число, которое преобразуется после кодирования в другое целое число. Известно, что для шифрования использовались описанные выше операции со следующими ограничениями:

1) каждая операция использовалась не более одного раза;

2) в маске число нулей не больше числа единиц;

3) в маске нет трех нулей подряд.

Предложите алгоритм шифрования, который преобразует число 81 в число 202, и соответствующий алгоритм дешифрования. Возможно ли, что существует число, которое не меняется после предложенного шифрования? Если да, укажите такое число, иначе обоснуйте ответ.

Решение задачи 6

Будем использовать следующие обозначения:

$\bar{X}$	Инверсия двоичного числа X
$X \oplus Y$	Сумма по модулю два двоичных чисел X и Y
$X \ll 1$	Циклический сдвиг влево двоичного числа X на одну позицию
$X \gg 1$	Циклический сдвиг вправо двоичного числа X на одну позицию

Заметим, что, если сдвиг и инверсия выполняются подряд, то неважно, в каком порядке они выполняются. Для побитового сложения по модулю два и сдвига порядок, безусловно, важен. Покажем это на примере. Пусть имеется число 11001011, и маска 10100111, выполним сдвиг и побитовое сложение в разном порядке.

Исходное число X	`11001011`
Сдвиг влево $X \ll 1$	`10010111`
Маска M	`10100111`

Исходное число X	`11001011`
Маска M	`10100111`
Сумма по модулю два	`01101100`

Сумма по модулю два $(X \ll 1) \oplus M$	00110000

$X \oplus M$	
Сдвиг влево $(X \oplus M) \ll 1$	11011000

Результаты получились разные, это объясняется тем, что до сдвига и после сдвига сложение по модулю два для заданного бита маски происходит с разными битами числа.

Как вычислить маску, если нам известно исходное число и результат? Если бит числа не меняется (0 в исходном числе и 0 в результате, 1 в исходном числе и 1 в результате), то в маске соответствующий бит равен нулю. Если же бит меняется (с 0 на 1 или с 1 на 0), его надо сложить по модулю два с единицей, т.е. соответствующий бит маски равен единице. Таким образом, маска представляет собой побитовое сложение по модулю два результата с исходным числом. В этом можно убедиться на примерах выше.

Из этого следует, что, если инвертировать исходное число, маска тоже инвертируется. Пусть какой-либо бит исходного числе равен соответствующему биту результата, тогда, как показано выше, соответствующий бит маски равен 0. После инверсии исходного числа его бит, наоборот, не равен соответствующему биту результата, следовательно, соответствующий бит маски должен быть равным 1. Пусть какой-либо бит исходного числа не равен соответствующему биту результата, тогда соответствующий бит маски равен 1. После инверсии исходного числа его бит, наоборот, равен соответствующему биту результата, следовательно, соответствующий бит маски должен быть равным 0. Это продемонстрировано на примере ниже.

Исходное число (X)	11001011
Его инверсия $(\bar{X})$	00110100
Маска (M)	10100111
Сумма по модулю два $(\bar{X} \oplus M)$	10010011

Исходное число (X)	11001011
Маска (M)	10100111
Инверсия маски $(\bar{M})$	01011000
Сумма по модулю два $(X \oplus \bar{M})$	10010011

Теперь подумаем, что происходит, если мы сначала складываем число с маской, а затем сдвигаем результат циклически влево? Например, мы сложили число X с маской M, получили число A, затем сдвинули его циклически влево, получили число B. Понятно, что A можно получить из B, если сдвинуть B циклически вправо. Ниже приведен пример.

Исходное число (X)	11001011
Маска (M)	10100111
Сумма по модулю два $(A = X \oplus M)$	01101100
Сдвиг влево $(B = A \ll 1)$	11011000

Таким образом, если имеется число X и результат R, у нас есть следующие возможности для поиска маски:

1. Вычисление маски как суммы по модулю два исходного числа с результатом, затем, возможно, инверсия маски ($M = X \oplus R$, $M = \overline{X \oplus R}$).

2. Сдвиг влево исходного числа, затем вычисление маски как суммы по модулю два исходного числа с результатом, затем, возможно, инверсия маски ($M = (X \ll 1) \oplus R$, $M = \overline{(X \ll 1) \oplus R}$).

3. Сдвиг вправо результата, зачем вычисление маски как суммы по модулю два числа с результатом, затем, возможно, инверсия маски ($M = X \oplus (R \gg 1)$, $M = \overline{X \oplus (R \gg 1)}$).

Нам остается подобрать маску, соответствующую заданным условиям.

Переведем сначала число и результат в двоичную систему счисления.

$$X = 81_{10} = 64 + 16 + 1 = 01010001_2$$

$$R = 202_{10} = 128 + 64 + 8 + 2 = 11001010_2$$

Способ 1.

X	01010001
R	11001010
$M = X \oplus R$	10011011
$M = \overline{X \oplus R}$	01100100

В маске $M = X \oplus R$ число нулей меньше числа единиц, нет трех нулей подряд. Значит, этот вариант подходит. Алгоритм шифрования: сложить исходное число с маской 10011011. Алгоритм дешифрования: сложить число с той же маской. При этом невозможно, чтобы число не изменилось после преобразования, поскольку биты исходного числа, соответствующие единичным битам маски, инвертируются. Вторая маска не подходит, так как число нулей больше числа единиц.

X	01010001
M	10011011
$X \oplus M$	11001010

Способ 2.

X	01010001
R	11001010

$X \ll 1$	10100010
$M = (X \ll 1) \oplus R$	01101000
$M = \overline{(X \ll 1) \oplus R}$	10010111

Маска 10010111 удовлетворяет заданным условиям. Алгоритм шифрования:

1. Сдвиг числа влево;
2. Инверсия;
3. Сложение по модулю два с маской 10010111.

X	01010001
$X \ll 1$	10100010
$\overline{X \ll 1}$	01011101
M	10010111
$\overline{(X \ll 1)} \oplus M$	11001010

Алгоритм дешифрования состоит в следующем:

1. Число R в двоичной системе счисления складываем по модулю два с маской 10010111, получаем число $\overline{X \ll 1}$;
2. Число $\overline{X \ll 1}$ инвертируем, получаем число $X \ll 1$.
3. Число $X \ll 1$ циклически сдвигаем вправо, получаем число X.

Пример для того же числа R показан в таблице ниже.

R	11001010
M	10010111
$R \oplus M = \overline{X \ll 1}$	01011101
$X \ll 1$	10100010
X	01010001

Рассмотрим теперь вопрос, возможно ли найти такое число, которое не изменится после шифрования. В общем виде двоичное число X можно записать как $X = x_7x_6x_5x_4x_3x_2x_1x_0$. Здесь x_i – разряды числа, которые принимают значение 0 и 1. Примем во внимание, что для любого $z \in \{0,1\}$ верно $z \oplus 0 = z$, $z \oplus 1 = \bar{z}$ (эти соотношения легко проверить построением таблиц истинности). Посмотрим, что происходит с числом X в ходе шифрования.

X	$x_7x_6x_5x_4x_3x_2x_1x_0$
$X \ll 1$	$x_6x_5x_4x_3x_2x_1x_0x_7$
$\overline{X \ll 1}$	$\bar{x}_6\bar{x}_5\bar{x}_4\bar{x}_3\bar{x}_2\bar{x}_1\bar{x}_0\bar{x}_7$
M	1 0 0 1 0 1 1 1
$R = (\overline{X \ll 1}) \oplus M$	$\bar{x}_6x_5x_4\bar{x}_3x_2\bar{x}_1\bar{x}_0\bar{x}_7$

Поскольку $R = X$, получаем уравнения для разрядов числа X

$$x_7 = \bar{x}_6; \qquad x_3 = x_2;$$
$$x_6 = x_5; \qquad x_2 = \bar{x}_1;$$
$$x_5 = x_4; \qquad x_1 = \bar{x}_0;$$
$$x_4 = \bar{x}_3; \qquad x_0 = \bar{x}_7.$$

Зафиксировав, например, $x_7 = 1$, получаем

$$1 = x_7 = \bar{x}_6; \qquad 1 = x_3 = x_2;$$
$$0 = x_6 = x_5; \qquad 1 = x_2 = \bar{x}_1;$$
$$0 = x_5 = x_4; \qquad 0 = x_1 = \bar{x}_0;$$
$$0 = x_4 = \bar{x}_3; \qquad 1 = x_0 = \bar{x}_7.$$

Из последнего уравнения получаем $x_7 = 0$. Система уравнений противоречива, т.е., не имеет решения. К такому же противоречию мы придем, если сначала зафиксируем $x_7 = 0$.

X	10001100
$X \ll 1$	00011001
$\overline{X \ll 1}$	11100110
M	01101010
$R = (\overline{X \ll 1}) \oplus M$	10001100

Способ 3.

X	01010001
R	11001010
$R \gg 1$	01100101
$M = X \oplus (R \gg 1)$	00110100
$M = \overline{X \oplus (R \gg 1)}$	11001011

Вторая маска соответствует условиям. Алгоритм шифрования может быть таким:

1. Сложение с маской 11001011.
2. Сдвиг влево.
3. Инверсия.

X	01010001
M	11001011
$X \oplus M$	10011010
$(X \oplus M) \ll 1$	00110101
$\overline{(X \oplus M) \ll 1}$	11001010

Алгоритм дешифрования:

1. Инверсия.
2. Сдвиг вправо.
3. Сложение с маской 11001011.

Ответ на вопрос, существует ли число, которое не меняется после шифрования, найдите самостоятельно. В ответе можно указать любой алгоритм.

Ответ. Алгоритм шифрования: сложить исходное число с маской 10011011. Алгоритм дешифрования: сложить число с той же маской. При этом невозможно, чтобы число не изменилось после преобразования, поскольку биты исходного числа, соответствующие единичным битам маски, инвертируются.

Задачи в других вариантах аналогичны, поэтому приводится решение только для одного варианта.

Задание 7. Системы счисления

Задача 7. Брон решил подготовиться к предстоящей поездке и обнаружил, что на новом снегоходе установленный электронный замок. Если он не отгадает код, невелика беда. Всегда есть старый и надежный снегоход, которым он пользовался ранее. Но ему так хотелось в ходе этой поездки на соседнюю нефтевышку опробовать новый аппарат, что он решил попытать удачу. К замку прилагается следующая записка:

Пароль – это количество цифр 3 в каждом числе. Но учти, что числа записаны в 10-й системе счисления, а тебе считать надо в 8-й.

285175 257 202 345

Ответ запишите в виде двоичного числа.

Решение задачи 7.

Сначала переведем все числа в 8 систему счисления

285 = 435

175 = 257

257 = 401

202 = 312

345 = 531

Далее всё что нам нужно сделать это подсчитать количество цифр 3 в получившихся числах

435 - 1

257 - 0

401 - 0

312 - 1

531 – 1

Запишем последовательно 0 и 1 и получим число двоичного вида.

Ответ: 1001

Задание 8. Булева алгебра

Задача 8. У Джона на столе лежала логическая схема и таблица истинности для данной схемы. Он умудрился пролить на таблицу кофе, в результате почти ничего не разобрать.

Вот что удалось восстановить, необходимо заполнить недостающие поля. $Z \to (X \vee Y \wedge Z) \equiv X$

...	**...**	**...**	**F**
1	**1**	**...**	**0**
1	**...**	**...**	**0**

Дописать недостающие строки и подписать столбцы

Решение задачи 8

Для начала построим таблицу истинности для заданной схемы $Z \to (X \vee Y \wedge Z) \equiv X$

x	y	z	$Y \wedge Z$	$(X \vee Y \wedge Z)$	$Z \to (X \vee Y \wedge Z)$	F
0	0	0	0	0	1	0
0	0	1	0	0	0	1
0	1	0	0	0	1	0
0	1	1	1	1	1	0
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	0	1	1	1
1	1	1	1	1	1	1

По заданной таблице нам нужны те строки, где F=0. Их получилось 3

x	y	z	F
0	0	0	0
0	1	0	0
0	1	1	0

Далее расставим столбцы так, чтобы таблица соответствовала предложенной в задании

y	**z**	**x**	**F**
1	**1**	**0**	**0**
1	**0**	**0**	**0**

Ответ: yzx

Задание 9. Кодирование и декодирование

Задача 9. Старый аппарат для передачи шифрованных сообщений был очень простым, но он хотя бы не давал слушать сообщения случайным радиолюбителям. Но совсем недавно сломался дешифратор, а починить его сразу нет времени. Имея ключи, получилось отсеять почти всю лишнюю информацию. Последний этап возлагают на вас, раскодируйте данные.

Для кодирования сообщений используется алфавит состоящий из 5 букв A,B,C,D,E и неравномерный двоичный код.

A = 111; B = 101; C = 100; D = 00; E = 01

Перехвачено 5 сообщений, но сигнал был довольно плохим и скорее всего в данных есть ошибки, нам кажется, что, хотя бы одно сообщение должно быть целым. Проверьте все и постарайтесь найти верное сообщение

111011001101111100

111011000111011100

111011000110111100

1110111000110111100

111011000110111110

Найти нужное сообщение, раскодировать его и записать ответ в виде ABCDE

Примечание: Имеется минимум одно верное сообщение.

Решение задачи 9.

Задача состоит в том, чтобы проверить можно ли из имеющего алфавита составить каждое из полученных сообщений.

Проверяем первое сообщение:

111 – 01 – 100 - 1101111100

Первый код будет 111, так как 11 – у нас нет такого набора в алфавите.

Затем мы можем попробовать два варианта 01 и 011, но 011 также нет в алфавите. Значит 01.

Следующий это 10 или 100. 10 – нет в алфавите, значит 100.

Следующий это 11 или 110. Но в алфавите нет обоих чисел.

Вывод: данное сообщение содержит ошибку, приступаем к следующему.

Проверяем второе сообщение:

111-01-100-01-110111100

После 01 не получается выделить значение из алфавита. Вывод: данное сообщение содержит ошибку.

Проверяем третье сообщение:

111-01-100-01-101-111-100

Данное сообщение не содержит ошибок и его можно расшифровать, подставив соответствующие буквы. AECEBAC

Проверяем четвертое сообщение:

111-01-110001101111100

После 01 не получается выделить значение из алфавита. Вывод: данное сообщение содержит ошибку.

Проверяем пятое сообщение:

111-01-100-01-101-111-10

В конце остается значение 10, а такого в алфавите нет. Вывод: данное сообщение содержит ошибку.

Ответ: AECEBAC

Задание 10. Количество информации

Задача 10. Дрон, делающий съёмки местности на месторождении Восточном, делает по 40 фотографий разрешением 1024 на 768 в минуту. Изображение использует палитру из 256 цветов, каждый отдельный пиксель кодируются минимально возможным количеством бит.

На какое время дрону хватит карты памяти размером 1350 Гб. Ответ представить в виде целого количества часов.

Решение задачи 10.

Для того чтобы посчитать минимально возможное количество бит для пикселя, нужно воспользоваться формулой: K = 2b, где K – количество различных цветов, b – количество битов для их кодирования. По нашим данным получается: 256 = 2b => b = 8.

Чтобы узнать сколько в минуту тратится бит, необходимо: 1024*768*8*40 = 251 658 240 бит/мин.

Ответ нужно представить в часах, поэтому 251 658 240*60 = 15 099 494 400 бит/ч.

Размер карты памяти в Гб, поэтому биты нужно перевести в Гб:

15 099 494 400 / (8*1024*1024*1024) = 1,7578125 Гб.

Теперь 1350/1,7578125 = 768 часов

Ответ: 768 часов.