Равномерный код: кодовые слова всех символов исходного алфавита имеют одинаковую длину.

Неравномерный код: кодовые слова имеют разную длину (например, Код Морзе).

Закодированное сообщение можно однозначно декодировать с начала, если выполняется *условие Фано*: *никакое кодовое слово не является началом другого кодового слова*. Закодированное сообщение можно однозначно декодировать с конца, если выполняется *обратное условие Фано*: никакое кодовое слово не является окончанием другого кодового слова

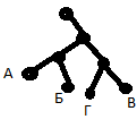
Префиксные коды: это коды со словами переменной длины, в котором ни одно кодовое слово не является началом другого кодового слова.

ПРИМЕР:

Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г и Д, решили использовать неравномерный двоичный код: А – 100 , Б – 101, В – 111, Г – 110. Укажите, каким кодовым словом из перечисленных ниже может быть закодирована буква Д. Код должен удовлетворять свойству однозначного декодирования. Если можно использовать более одного кодового слова, укажите кратчайшее из них. Варианты ответов: А) 1 Б) 0 В) 01 Г)10

РЕШЕНИЕ:

Построим дерево декодирования (0 - ветвь влево, 1 - ветвь вправо)



1 и 10 – не подходят (у этих точек есть ветви, а однозначное декодирование предполагает, что, проходя к каждой букве через узлы, мы не должны встретить другие буквы)

Подходит 0 и 01 (но в ответ пишем самое короткое 0). Ответ: Б

ФОРМУЛА ХАРТЛИ:

$$N = p^i$$

p – количество вариантов (используемых символов, сигналов), **p - МОЩНОСТЬ АЛФАВИТА**

i – количество информации (длина строки символов, длительность), **i - ДЛИНА КОДА**

N – число равновероятных событий, **N - КОЛИЧЕСТВО КОМБИНАЦИЙ**

ПРИМЕР:

Световое табло состоит из лампочек. Каждая лампочка может находиться в одном из трех состояний («включено», «выключено» или «мигает»). Какое наименьшее количество лампочек должно находится на табло, чтобы с его помощью можно было передать 18 различных сигналов?

$3^x = 18, x \approx 3$ Ответ: 3 лампочки

$$N = 2^i$$

N – мощность исходного алфавита

I – информационный вес одного символа алфавита (выражается в битах)

$$I = k \cdot i$$

k – количество символов в сообщении

i – информационный вес символа

I – информационный объем сообщения

ЕДИНИЦЫ ИЗМЕРЕНИЯ ИНФОРМАЦИИ:

- 1 байт = 2^3 (8) бит (это информационный вес символа алфавита мощностью 256)
- 1 Кбайт (Килобайт) = 2^{10} (1024) байт = 2^{13} (8192) бит
- 1 Мбайт (Мегабайт) = 2^{10} (1024) Кбайт = 2^{20} (1.048.576) байт = 2^{23}(8.388.608) бит
- 1 Гбайт (Гигабайт) = 2^{10} (1024) Мбайт = 2^{20}(1.048.576) Кбайт = 2^{30}(1.073.741.824) байт = 2^{33} (8.589.934.592) бит
- 1 Тбайт (Терабайт) = 2^{10} (1024) Гбайт = 2^{20} (1.048.576) Мбайт = 2^{30} (1.073.741.824) Кбайт = 2^{40} (1.099.511.627.776) байт = 2^{43} (8.796.093.022.208) бит
- 1 Пбайт (Петабайт) = 1024 Тбайта
- 1 Эбайт (Эксабайт) = 1024 Пбайта
- 1 Збайт (Зеттабайт) = 1024 Эбайта
- 1 Йбайт (Йоттабайт) = 1024 Збайта

Последовательность действий при переводе одних единиц измерения информации в другие приведена на схеме 1.

Схема 1

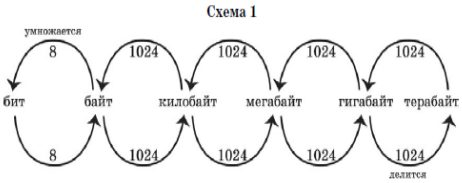


Таблица степеней двойки:

n	2^n
1	2^1=2
2	2^2=4
3	2^3=8
4	2^4=16
5	2^5=32
6	2^6=64
7	2^7=128
8	2^8=256
9	2^9=512
10	2^{10}=1024
11	2^{11}=2048
12	2^{12}=4096
13	2^{13}=8192

a^0=1

ПРИМЕР: Для регистрации на сайте некоторой страны пользователю требуется придумать пароль. Длина пароля – ровно 11 символов. В качестве символов используются десятичные цифры и 12 различных букв местного алфавита, причём все буквы используются в двух начертаниях: как строчные, так и заглавные (регистр буквы имеет значение!). Под хранение каждого такого пароля на компьютере отводится минимально возможное и одинаковое целое количество байтов, при этом используется посимвольное кодирование и все символы кодируются одинаковым и минимально возможным количеством битов. Определите объём памяти, который занимает хранение 60 паролей. Ответы:1) 540 байт 2) 600 байт 3) 660 байт 4) 720 байт

решение см. в соседней колонке

РЕШЕНИЕ:

Мощность алфавита = 10+12+12=34 (10 цифр, 12 заглавных и 12 строчных букв).

По формуле Хартли получаем:

2^i=34, откуда i=6 бит (т.к. 2^5 = 32 <34, 2^6 = 64>34).

Тогда I=11*6=66 бит (на 1 пароль) или примерно 9 байт (на 1 пароль)

И, окончательно, т.к.60 паролей, то 60*9=540 байт

Ответ: 1.

Существуют стандартные таблицы кодов. Они могут использовать один или два байта для кодирования одного символа.

Широко используется таблица кодов, известная как **стандарт ASCII** (**American Standart Code for Information Interchange** — Американский стандартный код для обмена информацией), использующая один байт для кодирования одного символа. ASCII представляет собой кодировку для представления десятичных цифр, символов латинского и национального алфавитов, знаков препинания, символов арифметических операций и управляющих символов. Управляющие символы называют непечатаемыми символами, к ним относятся такие, как «перевод строки» (код символа 10), «возврат каретки» (код 13) и др.

Первая половина кодовой таблицы содержит **стандартные** символы ASCII (символы с кодами 0 — 127), они одинаковые во всех странах.

В системе ASCII закреплены 2 таблицы кодирования – ***базовая и расширенная***. Базовая закрепляет значения кодов от 0 до 127, а расширенная относится к символам с номерами от 128 до 255.

Первые 32 кода базовой таблицы отданы производителям аппаратных средств. Начиная с кода 32 по код 127 размещены коды символов английского алфавита, знаков препинания, цифр, арифметических действий и некоторых вспомогательных символов. Коды с номерами от 128 до 255 используются для кодирования букв национального алфавита, символов национальной валюты и т.п. Существует несколько вариантов расширенной таблицы ASCII (примеры расширений: CP-866-используется для ОС MS-DOS, CP-1251-используется для Windows, КОИ-8-используется в сети Интернет, Unicode)

ЛОГИКА

ЛОГИЧЕСКАЯ ОПЕРАЦИЯ «НЕ»

Название: логическое отрицание (инверсия):
Обозначение в булевой алгебре: ¬A $\overline{A}$
Читается: частица НЕ
Обозначение в языках программирования:
not A (Паскаль, Бейсик) !A (Си)
Логическое отрицание истинное высказывание(1) превращает в ложное (0), а ложное высказывание (0) в истинное (1).

A	$\overline{A}$
0	1
1	0

Таблица истинности логической операции инверсия
(таблица истинности – это таблица, в которой слева перечисляются всевозможные значения исходного высказывания 0 или 1, а в правой части в последнем столбце записывают результат выполнения логической операции для каждого из этих вариантов)

ЛОГИЧЕСКАЯ ОПЕРАЦИЯ «И»

Название: логическое умножение (конъюнкция):
Обозначение в булевой алгебре: A&B A^B
(схожесть с алгеброй – значок пересечения ⋂)
Читается: союзы И (А, НО)
Обозначение в языках программирования:
A and B(Паскаль, Бейсик) A&&B (Си)
Логическая конъюнкция истинна тогда и только тогда, когда оба высказывания истинны (1) и ложна, когда ложно (0) хотя бы одно высказывание.

A	B	A·B
0	0	0
0	1	0
1	0	0
1	1	1

Таблица истинности логической операции конъюнкция

ЛОГИЧЕСКАЯ ОПЕРАЦИЯ «ИЛИ»

Название: логическое сложение (дизъюнкция):
Обозначение в булевой алгебре: A∨B
(схожесть с алгеброй – значок объединения ⋃)
Читается: союз ИЛИ
Обозначение в языках программирования:
A or B(Паскаль, Бейсик) A||B (Си)
Логическая дизъюнкция истинна тогда и только тогда, когда хотя бы одно высказывание истинно (1) и ложна, когда ложны (0) оба высказывания.

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

Таблица истинности логической операции дизъюнкция

ЛОГИЧЕСКАЯ ОПЕРАЦИЯ «ИСКЛЮЧАЮЩЕЕ ИЛИ»

Название: сложение по модулю 2 (разделительная дизъюнкция)
Обозначение в булевой алгебре: A⊕B A⊻B
Читается: оборот «или только..., или только ...»
Обозначение в языках программирования:
A xor B(Паскаль) A^B (Си)
Логическое сложение по модулю 2 истинно (1) тогда и только тогда, когда высказывания не равны и ложно (0), когда высказывания одинаковы.

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Таблица истинности логической операции исключающее или

ЛОГИЧЕСКАЯ ФУНКЦИЯ «ИМПЛИКАЦИЯ»

Название: импликация (следование)
Обозначение в булевой алгебре: A → B A ⇒ B
Читается: если А, то В; из А следует В;
для того, чтобы А, необходимо, чтобы В;
для того, чтобы В, достаточно, чтобы А
Логическая импликация ложна (0) тогда и только тогда, когда из истинного высказывания следует ложное (0), в остальных случаях импликация истинна (1).

A	B	A→B
0	0	1
0	1	1
1	0	0
1	1	1

Таблица истинности логической функции импликация

ЛОГИЧЕСКАЯ ФУНКЦИЯ «ЭКВИВАЛЕНТНОСТЬ»

Название: эквивалентность (равносильность)
Обозначение в булевой алгебре:
A↔B A⇔B A ≡ B A ⁓ B
Читается: А равносильно В; А тождественно равно В; для того, чтобы А, необходимо и достаточно В; А тогда и только тогда, когда В.
Логическая эквивалентность истинна (1) тогда и только тогда, когда высказывания одинаковы, и ложна (0), когда высказывания не равны.

A	B	A↔B
0	0	1
0	1	0
1	0	0
1	1	1

Таблица истинности логической функции эквивалентность (эквиваленция, равнозначность)

ЛОГИЧЕСКАЯ ОПЕРАЦИЯ «ШТРИХ ШЕФФЕРА»

Название: И-НЕ (штрих Шеффера)
Обозначение в булевой алгебре: A|B
(англ. nand – not and – отрицание конъюнкции)
Штрих Шеффера ложен (0) тогда и только тогда, когда оба высказывания истинны, и истинен (1) в остальных случаях.

Штрих Шеффера

A	B	A|B
0	0	1
0	1	1
1	0	1
1	1	0

$A|B = \overline{A \cdot B}$

Таблица истинности логической операции Штрих Шеффера

ЛОГИЧЕСКАЯ ОПЕРАЦИЯ «СТРЕЛКА ПИРСА»

Название: ИЛИ-НЕ (стрелка Пирса)
Обозначение в булевой алгебре: A↓B или A↑B
(англ. nor – not or – отрицание дизъюнкции)
Стрелка Пирса истинна (1) тогда и только тогда, когда оба высказывания ложны, и ложна (0) в остальных случаях.

Стрелка Пирса

A	B	A↓B
0	0	1
0	1	0
1	0	0
1	1	0

$A \downarrow B = \overline{A + B}$

Таблица истинности логической операции стрелка Пирса

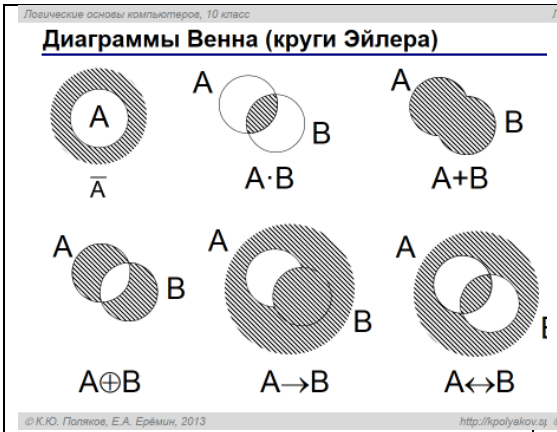
Логические переменные		Логические операции					
		отрицание	конъюнкция	дизъюнкция	исключающая дизъюнкция	импликация	эквиваленция
A	B	¬A	A & B	A ∨ B	A ⊕ B	A → B	A ↔ B
0	0	1	0	0	0	1	1
0	1	1	0	1	1	1	0
1	0	0	0	1	1	0	0
1	1	0	1	1	0	1	1

ПРИОРИТЕТ ВЫПОЛНЕНИЯ ЛОГИЧЕСКИХ ОПРЕЦИЙ:

- 1) В СКОБКАХ
- 2) НЕ
- 3) И
- 4) ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ
- 6) ИМПЛИКАЦИЯ
- 7) ЭКВИВАЛЕНТНОСТЬ

ЛОГИЧЕСКИЕ ЗАКОНЫ:

1) ¬¬A=A Закон двойного отрицания
2) A∨ A=A A&A=A
Законы идемпотентности
3) A ∨ 0=A A ∨ 1=1 A&0=0 A&1=A
(Законы исключения констант)
4) A ∨ (A&B) = A A& (A ∨ B) = A
(Законы поглощения)
5) A ∨ ¬A = 1 (Закон исключения третьего)
A & ¬A = 0 (Закон непротиворечия)
6) A & B = B & A A ∨ B = B ∨ A
(Законы коммутативности или переместительные законы)
7) A & (B& C) = (A & B) & C = A & B & C
A ∨(B ∨ C) = (A ∨ B) ∨ C =A ∨ B ∨ C
(Законы ассоциативности или сочетательные законы)
8) ¬ (A ∨ B) = ¬A & ¬B
¬ (A & B) = ¬A ∨ ¬B
(Законы де Моргана – названы в честь шотландского математика и логика де Моргана)
9) A & (B ∨ C) = (A & B) ∨ (A & C)
A ∨ (B & C)= (A ∨ B) & (A ∨ C)
(Законы дистрибутивности или распределительные законы)
10) A → B = ¬A ∨ B (закон снятия импликации)
11) A ↔ B = (A → B) & (B → A) =
=(A & B) ∨ (¬A & ¬B) = (A ∨¬B)&(¬A ∨B)
(закон снятия эквивалентности)
12) A ⊕ B = (A&¬B) ∨ (¬A&B)
(закон снятия сложения по модулю 2)



ПРИМЕР: Известно количество сайтов, которые находит поисковый сервер по следующим запросам: Огурцы – 100 сайтов, Помидоры – 200 сайтов, Огурцы & Помидоры – 50 сайтов. Найти, сколько сайтов будет найдено по запросу Огурцы | Помидоры?

О | П = О + П – О&П =

= 100+200-50=250

Ответ: 250.

ФОРМУЛА ДЛЯ ВЫЧИСЛЕНИЯ информационного объема звукового файла

$I = i \bullet \omega \bullet t \bullet r$

i – глубина кодирования звука (в битах), бит

ω – частота (в ГГерцах, Гц

t – время звучания, длительность (в секундах), с

r – режим (ммон- 1, стерео - 2, квадро - 4 и тт.д.

I – информационный объем звукового файла (в битах)

Глубина кодирования звука

- это количество информации, которое необходимо для кодирования дискретных уровней громкости цифрового звука.

Если известна глубина кодирования, то количество уровней громкости цифрового звука можно рассчитать по формуле $N = 2^I$

Пусть глубина кодирования звука составляет 16 битов, тогда количество уровней громкости звука равно: $N = 2^I = 2^{16} = 65\,536$.

ФОРМУЛА ДЛЯ ВЫЧИСЛЕНИЯ информационного объема видеофайла

$I_{видео} = (I_{звука} + I_{графики}) /$ степень сжатия

$I_{звука} = i \bullet \omega \bullet t \bullet r$

i – глубина кодировани я звука (в битах), бит

ω – частота (в ГГерцах , Гц

t – время звучания, длительнос ть (в секундах), с

r – режим (ммон - 1, стерео - 2, квадро - 4 и тт.д.

$I_{звука}$ – информацио нный объем звукового файла (в битах)

$I_{графики} = i \bullet a \bullet b \bullet v \bullet t$

i – глубина кодировани я цвета (в битах), бит

a – высота картинки в пикселях

b – ширина картинки в пикселях

v – скорость кадров (кколичест во кадров в секунду)

t – время, длительнос ть видеофайла (в секундах), с

$I_{графики}$ - информацио нный объем картинок

ЛИНЕЙНЫЙ АЛГОРИТМ –

- это алгоритм, в котором команды выполняются последовательно одна за другой

БЛОК-СХЕМА ЛИНЕЙНОГО АЛГОРИТМА

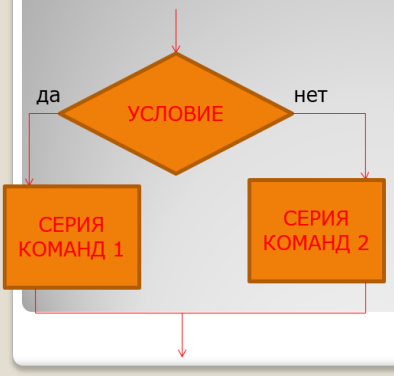
ЛИНЕЙНЫЙ АЛГОРИТМ НА PASCAL

```
Program <ИМЯ ПРОГРАММЫ> ;
Var <СПИСОК ПЕРЕМЕННЫХ> : <ТИП ПЕРЕМЕННЫХ> ;
Begin
<КОМАНДА ВВОДА>  (НАПРИМЕР, Readln)
<КОМАНДА ПРИСВАИВАНИЯ 1>
<КОМАНДА ПРИСВАИВАНИЯ 2>
. . . . . .
<КОМАНДА ПРИСВАИВАНИЯ n>
<КОМАНДА ВЫВОДА>  (НАПРИМЕР, Writeln)
End .
```

ВЕТВЯЩИЙСЯ АЛГОРИТМ –

- это алгоритм, в котором выполняется одна или другая серия команд в зависимости от истинности или ложности условия

ФРАГМЕНТ БЛОК-СХЕМЫ ВЕТВЯЩЕГОСЯ АЛГОРИТМА



ФРАГМЕНТ ВЕТВЯЩЕГОСЯ АЛГОРИТМА НА PASCAL

```
If <УСЛОВИЕ>  then
 begin
 <СЕРИЯ КОМАНД 1>
 end
  Else
   begin
   <СЕРИЯ КОМАНД 2>
   end;
```

АЛГОРИТМИЧЕСКАЯ СТРУКТУРА «ВЫБОР» –

- это алгоритм, в котором выполняется одна из нескольких последовательностей команд при истинности соответствующего условия (удобно использовать вместо вложенного ветвления IF)

ФРАГМЕНТ БЛОК-СХЕМЫ АЛГОРИТМА ВЫБОР

ФРАГМЕНТ АЛГОРИТМА «ВЫБОР »НА PASCAL

```
Case <ВЫРАЖЕНИЕ> Of
<ВАРИАНТ1> : <ОПЕРАТОР 1> ;
<ВАРИАНТ2> :<ОПЕРАТОР 2> ;
. . . . . .
<ВАРИАНТ N> :<ОПЕРАТОР N-1 >;
Else <ОПЕРАТОР N>
End;
```

АЛГОРИТМИЧЕСКАЯ СТРУКТУРА «ЦИКЛ» –

- это алгоритм, в котором серия команд (она называется телом цикла) выполняется многократно.
- Тело цикла определяет ЧТО повторять, а заголовок цикла (вид цикла) определяет СКОЛЬКО РАЗ повторять.

ВИДЫ ЦИКЛОВ

Безусловный цикл

Условный цикл

Цикл со счетчиком (цикл n раз)

Цикл с условием

Цикл с предусловием

Цикл с постусловием

Цикл со счетчиком (цикл n раз) Используется, когда заранее известно, сколько раз необходимо выполнить цикл.

ФРАГМЕНТ АЛГОРИТМА ЦИКЛА n РАЗ НА АЯ (алгоритмическом языке)

```
НЦ <ПАРАМЕТР ЦИКЛА>  ОТ <НАЧАЛЬНОЕ ЗНАЧЕНИЕ> ДО <КОНЕЧНОЕ ЗНАЧЕНИЕ> ШАГ h
ПОВТОРЯТЬ
<СЕРИЯ КОМАНД – ТЕЛО ЦИКЛА >
КЦ
```

ФРАГМЕНТ АЛГОРИТМА ЦИКЛА n РАЗ НА PASCAL

```
For <СЧЕТЧИК> : = <НАЧАЛЬНОЕ ЗНАЧЕНИЕ> to (или downto) <КОНЕЧНОЕ ЗНАЧЕНИЕ> do
begin
<СЕРИЯ КОМАНД – ТЕЛО ЦИКЛА >
End;
```

Цикл с предусловием Используется, когда заранее неизвестно, сколько раз необходимо выполнить цикл.

ФРАГМЕНТ АЛГОРИТМА ЦИКЛА с *предусловием* (условие впереди) НА АЯ (алгоритмическом языке)

```
НЦ ПОКА <УСЛОВИЕ>
ПОВТОРЯТЬ
<СЕРИЯ КОМАНД – ТЕЛО ЦИКЛА >
КЦ
```

Данный вид цикла может не выполниться ни разу. Ответьте, почему?

ФРАГМЕНТ АЛГОРИТМА ЦИКЛА С ПРЕДУСЛОВИЕМ НА PASCAL
ЦИКЛ С ИСТИННЫМ ПРЕДУСЛОВИЕМ

```
While <УСЛОВИЕ>  Do
Begin
<СЕРИЯ КОМАНД – ТЕЛО ЦИКЛА >
Тело цикла выполняется пока условие истинно
End;
```

Цикл с постусловием Используется, когда заранее неизвестно, сколько раз необходимо выполнить цикл.

ФРАГМЕНТ АЛГОРИТМА ЦИКЛА с *постусловием* (*условие после*) НА АЯ (алгоритмическом языке)

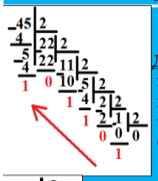
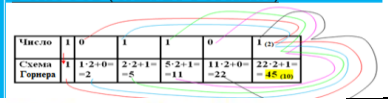
```
НЦ
ПОВТОРЯТЬ
<СЕРИЯ КОМАНД – ТЕЛО ЦИКЛА >
КЦ  ПОКА <УСЛОВИЕ>
```

Данный вид цикла выполнится хотя бы один раз. Ответьте, почему?

ФРАГМЕНТ АЛГОРИТМА ЦИКЛА С ПОСТУСЛОВИЕМ НА PASCAL
ЦИКЛ С ЛОЖНЫМ ПОСТУСЛОВИЕМ

```
Repeat
<СЕРИЯ КОМАНД – ТЕЛО ЦИКЛА >
Тело цикла выполняется пока условие
Ложно
Until <УСЛОВИЕ> ;
```

Перевод чисел с использованием 10-ой системы счисления

Перевод чисел из десятичной системы счисления в систему счисления с основанием n ИЗ 10-ОЙ	Перевод чисел из системы счисления с основанием n в десятичную систему счисления В 10-УЮ
 ЦЕЛЫЕ ЧИСЛА 1 СПОСОБ (ОБЫЧНОЕ ДЕЛЕНИЕ): осуществляют последовательное деление на n до тех пор, пока частное не станет = 0, остатки записывают в обратном порядке.	ЦЕЛЫЕ ЧИСЛА 1 СПОСОБ (ПО СТЕПЕНЯМ): I) Над каждой цифрой справа налево ставим целые числа в порядке возрастания 0,1,2…; II) Составляем сумму произведений цифр числа на соответствующие степени основания (показатели степени – числа из п.I) $101101_{(2)} = 1\cdot2^5 + 0\cdot2^4 + 1\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 = 32+0+8+4+0+1 = 45_{(10)}$
 ЦЕЛЫЕ ЧИСЛА 2 СПОСОБ (КОРОТКОЕ ДЕЛЕНИЕ) $45_{(10)} = 101101_{(2)}$	ЦЕЛЫЕ ЧИСЛА 2 СПОСОБ (СХЕМА ГОРНЕРА) 

$0{,}3125_{(10)} = 0{,}24_{(8)}$

$0{,}3125 \cdot 8 = 2{,}5$

$0{,}5 \cdot 8 = 4{,}0$

$0 \cdot 8 = 0$

$0{,}36_{(10)} = 0{,}23432\ldots_{(7)}$

$= 0{,}(2343)_{(7)}$

$0{,}36 \cdot 7 = 2{,}52$

$0{,}52 \cdot 7 = 3{,}64$

$0{,}64 \cdot 7 = 4{,}48$

$0{,}48 \cdot 7 = 3{,}36$

$0{,}36 \cdot 7 = 2{,}52$

и т.д.....

Перевод чисел с использованием 10-ой системы счисления

Перевод чисел из десятичной системы счисления в систему счисления с основанием n. ИЗ 10-ОЙ	Перевод чисел из системы счисления с основанием n в десятичную систему счисления. В 10-УЮ
ДРОБНЫЕ ЧИСЛА 1 СПОСОБ (КОГДА ЦЕЛОЙ ЧАСТИ НЕТ): осуществляем последовательное умножение на n, причем очередному умножению подвергается дробная часть результата, ответ записывается в виде целых частей в порядке их появления (дробь может оказаться и периодической).	ДРОБНЫЕ ЧИСЛА 1 СПОСОБ(КОГДА ЦЕЛОЙ ЧАСТИ НЕТ) I) Над каждой цифрой слева направо ставим целые числа в порядке убывания -1,-2…; II) Составляем сумму произведений цифр числа на соответствующие степени основания (показатели оснований – из п.I)
ДРОБНЫЕ ЧИСЛА 2 СПОСОБ (КОГДА ЦЕЛАЯ ЧАСТЬ ЕСТЬ): Переводим отдельно целую часть , отдельно дробную и отделяем их в ответе запятой.	ДРОБНЫЕ ЧИСЛА 2 СПОСОБ (КОГДА ЦЕЛАЯ ЧАСТЬ ЕСТЬ) I) Над каждой цифрой ставим 0,1,2,… (влево над целой частью) и -1,-2… (вправо над дробной частью) II) Составляем сумму произведений цифр числа на соответствующие степени основания (показатели оснований – из п.I)

$194_{(10)} = C2_{(16)}$

194 | 16

12 | 2

0 | 12=C

$0{,}15625_{(10)} = 0{,}28_{(16)}$

$0{,}15625 \cdot 16 = 2{,}5$

$0{,}5 \cdot 16 = 8{,}0$

$0 \cdot 16 = 0$

ТОГДА $194{,}15625_{(10)} = C2{,}28_{(16)}$

$$0{,}28_{(16)} = 2\cdot16^{-1} + 8\cdot16^{-2} = \frac{1}{8} + \frac{1}{32} = \frac{5}{32} = 0{,}15625_{(10)}$$

$$C2{,}28_{(16)} = 12\cdot16^{1} + 2\cdot16^{0} + 2\cdot16^{-1} + 8\cdot16^{-2} = 192 + 2 + \frac{1}{8} + \frac{1}{32} = 194\frac{5}{32} = 194{,}15625_{(10)}$$

Перевод чисел без использования 10-ой системы счисления

Перевод чисел из ДВОИЧНОЙ системы счисления в ВОСЬМЕРИЧНУЮ систему счисления. 2→8	Перевод чисел из ВОСЬМЕРИЧНОЙ системы счисления в ДВОИЧНУЮ систему счисления. 8→2
ЦЕЛЫЕ ЧИСЛА 1 СПОСОБ (ТРИАДЫ-БЕЗ 10-ОЙ С.С.) Число разбивают справа налево на группы из 3 цифр (триады) и каждую триаду переводят в 8-ую систему счисления (при необходимости слева дописывают нули).	ЦЕЛЫЕ ЧИСЛА 1 СПОСОБ (ТРИАДЫ-БЕЗ 10-ОЙ С.С.) Каждую цифру числа представляют в виде триады и записывают последовательно одна за другой (при необходимости в ответе слева незначащие нули убирают).
ЦЕЛЫЕ ЧИСЛА 2 СПОСОБ (ИСПОЛЬЗУЯ ДВА ПЕРЕХОДА: СНАЧАЛА ПЕРЕВЕСТИ ИЗ 2-ОЙ В 10-УЮ, А ЗАТЕМ ИЗ 10-ОЙ В 8-УЮ)	ЦЕЛЫЕ ЧИСЛА 2 СПОСОБ (ИСПОЛЬЗУЯ ДВА ПЕРЕХОДА: СНАЧАЛА ПЕРЕВЕСТИ ИЗ 8-ОЙ В 10-УЮ, А ЗАТЕМ ИЗ 10-ОЙ В 2-УЮ)

Перевод чисел без использования 10-ой системы счисления

Перевод чисел из ДВОИЧНОЙ системы счисления в ШЕСТНАДЦАТЕРИЧНУЮ систему счисления. 2→16	Перевод чисел из ШЕСТНАДЦАТЕРИЧНОЙ системы счисления в ДВОИЧНУЮ систему счисления. 16→2
ЦЕЛЫЕ ЧИСЛА 1 СПОСОБ (ТЕТРАДЫ-БЕЗ 10-ОЙ С.С.) Число разбивают справа налево на группы из 4 цифр (тетрады) и каждую тетраду переводят в 16-ую систему счисления (при необходимости слева дописывают нули).	ЦЕЛЫЕ ЧИСЛА 1 СПОСОБ (ТЕТРАДЫ-БЕЗ 10-ОЙ С.С.) Каждую цифру числа представляют в виде тетрады и записывают последовательно одна за другой (при необходимости в ответе слева незначащие нули убирают).
ЦЕЛЫЕ ЧИСЛА 2 СПОСОБ (ИСПОЛЬЗУЯ ДВА ПЕРЕХОДА: СНАЧАЛА ПЕРЕВЕСТИ ИЗ 2-ОЙ В 10-УЮ, А ЗАТЕМ ИЗ 10-ОЙ В 16-УЮ)	ЦЕЛЫЕ ЧИСЛА 2 СПОСОБ (ИСПОЛЬЗУЯ ДВА ПЕРЕХОДА: СНАЧАЛА ПЕРЕВЕСТИ ИЗ 16-ОЙ В 10-УЮ, А ЗАТЕМ ИЗ 10-ОЙ В 2-УЮ)

ТРИАДЫ	
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

ТЕТРАДЫ			
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

Алгоритм получения дополнительного кода. Для получения дополнительного кода отрицательного числа можно использовать довольно простой алгоритм:

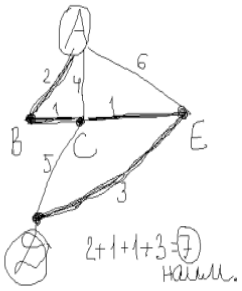
1. Модуль числа записать прямым кодом в *n* двоичных разрядах.
2. Получить обратный код числа, для этого значения всех битов инвертировать (все единицы заменить на нули и все нули заменить на единицы).
3. К полученному обратному коду прибавить единицу.

МОДЕЛИРОВАНИЕ (ПОИСК КРАТЧАЙШЕГО ПУТИ ПО ТАБЛИЦЕ)

Определите кратчайший путь между пунктами A и D.

Способ решения: с помощью схемы

	A	B	C	D	E
A		2	4		6
B	2		1		
C	4	1		5	1
D			5		3
E	6		1	3	



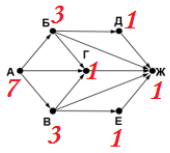
МОДЕЛИРОВАНИЕ (ПОИСК КОЛИЧЕСТВА ПУТЕЙ НА ГРАФЕ)

42

Количество путей

Сколько существует различных путей из А в Ж?

Способ решения – в виде таблицы (сначала перечисляем все вершины с конца в начало, затем заполняем строчку – куда идем из этих вершин, а затем считаем количество путей для каждой вершины, в последнем столбце – ответ).



Вершины	Ж	Е	Д	Г	В	Б	А
Куда идем ?	Ж	Ж	Ж	Ж	Г Ж Е	Д Ж Г	Б В Г
Количество путей	1	1	1	1	3	3	7

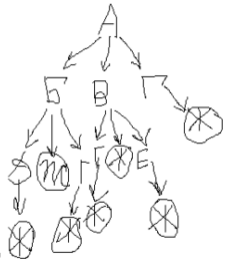
Ответ: 7

МОДЕЛИРОВАНИЕ (ПОИСК КОЛИЧЕСТВА ПУТЕЙ НА ГРАФЕ)

43

2 способ решения – с помощью построения дерева решений и далее считаем количество кружочков (буква Ж обведена в кружочек)

ДЕРЕВО РЕШЕНИЙ

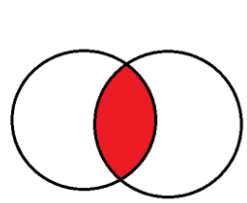


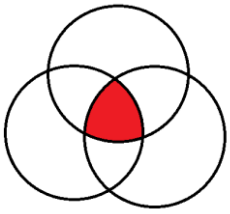
3 один способ решения - вручную, водим с помощью карандаша по линиям (самый плохой способ, ненадежный)

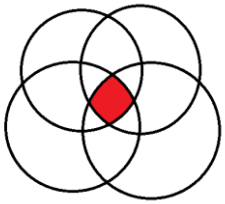
Информация для Черепашки:

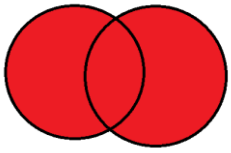
1) угол правильного (равностороннего) треугольника равен 60^0
2) угол правильного четырехугольника (квадрата) равен 90^0
3) угол правильного пятиугольника 108^0
4) угол правильного шестиугольника 120^0
5) угол правильного восьмиугольника 135^0
6) угол правильного девятиугольника 140^0
7) угол правильного десятиугольника 144^0
8) угол правильного двенадцатиугольника 150^0
9) 36^0 Пятилучевая звезда
10) 45^0 Восьмилучевая звезда

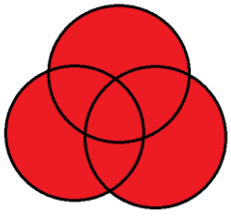
КРУГИ ЭЙЛЕРА

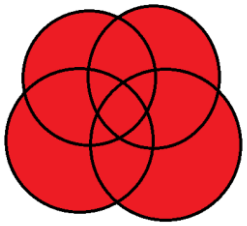

A&B

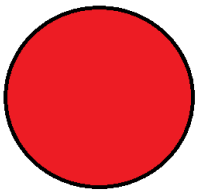

A&B&C

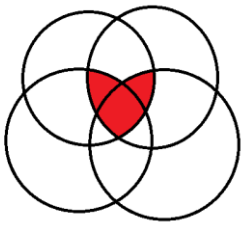

A&B&C&D

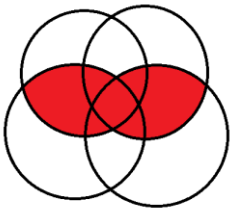

A|B

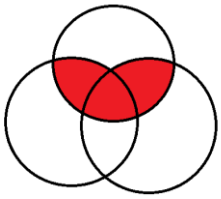

A|B|C

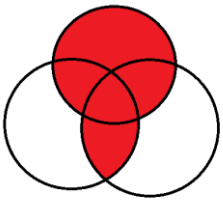

A|B|C|D

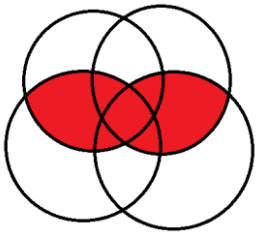

A


A&B&(C|D)


(A&B)|(C&D)


A&(B|C) = (B|C)&A


A|(B&C) = (B&C)|A


(A|B)&(C|D)

Адрес документа в Интернете (URL = *Uniform Resource Locator*) состоит из следующих частей:

- протокол, чаще всего **`http`** (для Web-страниц) или **`ftp`** (для файловых архивов)
- знаки **`://`**, отделяющие протокол от остальной части адреса
- доменное имя (или IP-адрес) сайта
- каталог на сервере, где находится файл
- имя файла

ТАКИМ ОБРАЗОМ:

Адрес файла в сети Интернет: протокол://сервер/файл или протокол://сервер/каталог/файл

протокол **://** доменное имя (или IP-адрес) сайта (сервер) / каталог на сервере, где находится файл / имя файла

Пример: http://www.vasya.ru/home/user/vasya/qu-qu.zip - здесь задан протокол http, доменное имя сайта (сервер) www.vasya.ru, каталог на сайте home/user/vasya и имя файла qu-qu.zip.

Каждый компьютер, подключенный к сети Интернет, должен иметь собственный адрес, который называют IP-адресом (IP = *Internet Protocol*). IP-адрес состоит из четырех чисел, разделенных точками; каждое из этих чисел находится в интервале 0…255, например: **192.168.85.210.**