1. Определение высказывания.

Высказывание - утверждение, о котором можно сказать, истинно оно, или ложно.

2. Арность операции (функции).

Арность операции (функции) - количество аргументов данной операции (функции).

3. Булева алгебра (определение + аксиомы/законы только перечислить).

Булева алгебра - множество B, состоящее из элементов 0, 1, на котором заданы бинарные операции конъюнкции и дизъюнкции, и унарная операция отрицания следующие аксиомы (законы) для всех a, b, c из B:

1. Закон коммутативности

$$a \wedge b = b \wedge a$$
$$a \vee b = b \vee a$$

2. Закон дистрибутивности

$$(a \wedge b) \vee c = (a \vee c) \wedge (b \vee c)$$
$$(a \vee b) \wedge c = (a \wedge c) \vee (b \wedge c)$$

3. Закон ассоциативности

$$(a \vee b) \vee c = a \vee (b \vee c)$$
$$(a \wedge b) \wedge c = a \wedge (b \wedge c)$$

4. Закон тождества

$$a \vee 1 = 1$$
$$a \wedge 1 = a$$
$$a \vee 0 = a$$
$$a \wedge 0 = 0$$

5. Закон дополнения

$$a \vee \neg a = 1$$
$$a \wedge \neg a = 0$$

4. Булево множество.

Булево множество - множество, состоящее только из элементов 0, 1.

5. Закон единственности дополнения.

Дополнение произвольного элемента x единственным образом определяется его свойствами:

$x \vee \neg x = 1$ $\quad x \vee x* = 1$ Тогда $\neg x = x*$
$x \wedge \neg x = 0$ $\quad x \wedge x* = 0$

6. Закон инволюции.

$$\neg(\neg x) = x$$

7. Теорема склеивания.

$$(x \wedge y) \vee x = x$$
$$(x \vee y) \wedge x = x$$

8. В чем отличие функции и формулы?

Функция - правило, согласно которому каждому элементу из множества X ставится в соответствие единственный элемент из множества Y.
Для булевых функций X = Y = {0; 1}
У одной функции может быть несколько формул.

9. Число булевых функций n аргументов.

$$2^{2^n}$$

10. ДНФ.

Дизъюнктивная нормальная форма - дизъюнкция выражений, которые могут являться:
1. Отдельным аргументом (возможно с инверсией)
2. Простым конъюнктом

Ex.
ДНФ:
$AB + BC$

не ДНФ:
$AB + B * (A + C)$

11. КНФ.

Конъюнктивная нормальная форма - конъюнкция выражений, которые могут являться:
1. Отдельным аргументом (возможно с инверсией)
2. Простым дизъюнктом

Ex.
КНФ:
$(A + B)(C + B)$

не КНФ:
$(A + B)(C + AB)$

12. Минтерм.

Минтерм - булева функция, принимающая единичное значение только на одном наборе переменных.

13. СДНФ.

Совершенная дизъюнктивная нормальная форма - представление функции в виде дизъюнкции минтермов n аргументов.

14. Импликанта.

Имликанта g функции f - функция, множество минтермов которой является подмножеством множества минтермов функции f.

15. Простая импликанта.

Простая импликанта - импликанта, состоящая из одной простой конъюнкции, такой, что никакая ее часть не является импликантой функции.

16. Число импликант функции.

Число импликант функции определяется как 2^m, где m - число минтермов в записи функции в СНДФ.

17. Минимизация булевой формулы. Методы минимизации.

Минимизация булевой формулы - нахождение наименьшего числа простых импликант, дизъюнкция которых описывает исходную функцию.
Методы минимизации:
1. Алгебраический
2. Метод Квайна
3. Метод Квайна-Макласки
4. Метод Петрика
5. Карты Вейча/Карно

18. Сокращенная ДНФ.

Сокращенная ДНФ - запись функции, в которой:
1. Любые два слагаемых отличаются минимум в двух местах
2. Ни один из конъюнктов не содержится в другом

19. Тупиковая ДНФ.

Тупиковая ДНФ - сокращенная ДНФ, не имеющая лишних (не влияющих на таблицу истинности) имликант.

20. Минимальная ДНФ.

Минимальная ДНФ - тупиковая ДНФ, содержащая минимальное число вхождений переменных.

21. Макстерм.

Макстерм - булева функция, принимающая единичные значения на всех наборах, кроме одного.

22. СКНФ.

Совершенная конъюнктивная нормальная форма - представление функции в виде конъюнкции n аргументов.

23. Теорема разложения для ДНФ/КНФ.

Всякую булеву функцию можно представить в виде:

$$f(x_1, x_2, \dots, x_n) = x_i \cdot f(x_1, \dots, 1, \dots, x_n) \vee \overline{x_i} \cdot f(x_1, \dots, 0, \dots, x_n)$$

$$f(x_1, x_2, \dots, x_n) = (x_i \vee f(x_1, \dots, 1, \dots, x_n)) \wedge (\overline{x_i} \vee f(x_1, \dots, 0, \dots, x_n))$$

24. Полином Жегалкина. Методы построения.

Полином Жегалкина - полином с коэффициентами 0 и 1, в котором произведение представлено операцией конъюнкции, а сложение - исключающим или (сложением по модулю 2).

Методы построения:
1. Метод треугольника
2. Метод Паскаля

25. Суперпозиция функций.

Суперпозиция функций - функция, полученная из некоторого множества функций путем подстановок одной функции в другую и/или отождествлением переменных.

26. Подстановка функции g в функцию f.

Подстановка функции g в функцию f - замена i-го аргумента функции f на функцию g.

27. Отождествление переменных в функции f.

Отождествление переменных функции f - замена i-го аргумента функции f на аргумент j.

28. Замкнутое множество функций.

Замкнутое множество функций - множество, в котором любая функция является суперпозицией некоторого подмножества функций данного множества.

29. Замыкание множества функций.

Замыкание множества функций A - некоторое подмножество булевых функций, такое что любую из этих функций можно выразить через функции множества A.

30. Полная система функций.

Полная система функций - множество функций, для которой замыкание совпадает с множеством всех булевых функций.

31. Безызбыточная полная система функций / Избыточная полная система функций.

Безызбыточная полная система функций - полная система функций, которая перестает быть таковой после исключения из нее любой функции.
Полная система функций является избыточной, если из нее можно исключить одну или несколько функций, получив безызбыточную полную систему.

32. Самодвойственная функция. Число самодвойственных функций n аргументов.

Самодвойственная функция - функция, принимающая противоположные значения на противоположных наборах аргументов.
Число самодвойственных функций n аргументов:

33. Линейная функция. Число линейных функций n аргументов.

Линейная функция - функция, не имеющая конъюнкций в своем представлении в виде полинома Жегалкина.
Число линейных функций n аргументов:

34. Монотонная функция.

Монотонная функция - функция, не убывающая на сравнимых наборах.
Число монотонных функций n аргументов:

35. Сравнимые наборы.

Наборы a и b называются сравнимыми, если

36. Функции, сохраняющие константу. Число таких функций n аргументов.

Функция, сохраняющая единицу - функция, принимающая единичное значение на единичном наборе аргументов.

Функция, сохраняющая ноль - функция, принимающая нулевое значение на нулевом наборе аргументов.

37. Формулировка критерия Поста.

Система функций полная тогда и только тогда, когда она целиком не содержится ни в одном из предполных классов.
Другими словами: среди функций полной системы найдется хоть одна:
1. Не сохраняющая единицу
2. Не сохраняющая ноль
3. Несамодвойственная
4. Нелинейная
5. Немонотонная

Пояснение:
Класс функций - множество булевых функций, объединенное некоторым свойством.
Предполный класс функций - замкнутое множество булевых функций, которое:
1. не является полной системой,

2. является максимальным по включению среди всех замкнутых множеств, не являющихся полными системами.

38. Перечислить полные системы из одной функции.

{стрелка Пирса}, {штрих Шеффера}

39. Частично определенная (недоопределенная) функция.

Частично определенная функция - функция, значение которой определено не на всех наборах.

40. Кодирование.

Код - система условных знаков для представления информации.

Кодирование - перевод информации с помощью некоторого кода в форму, удобную для обработки, передачи и хранения.

Декодирование - процесс восстановления содержания закодированной информации.

Типы кодирования:
1. Кодирование с однозначным декодированием (получатель может однозначно декодировать сообщение)
2. Кодирование с минимальной избыточностью (при кодировании нет значительного увеличения объема)
3. Надежное кодирование (устойчивое к искажениям)

41. Однозначно декодируемый код.

Однозначно декодируемый код - код, допускающий один способ декодирования.

42. Равномерный код.

Равномерный код - код, в котором все кодовые слова попарно различны, но имеют одинаковую длину.

43. Энтропия сообщения.

Энтропия сообщения - количество информации, которое в среднем приходится на один символ. (Нижняя граница средней длины сообщения).

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

, где p_i - вероятность i-го события (символа).

44. Неравенство Крафта-Макмиллана.

Необходимое условие декодируемости: если схема алфавитного кодирования однозначно декодируема, то выполняется неравенство Крафта-Макмиллана:

$$\sum_{i=1}^{\mathrm{r}} 2^{-l_i} \leq 1$$

, где r - количество букв в исходном алфавите, q - количество букв в кодирующем алфавите, а l_1, l_2, …, l_r - длины кодовых слов.

45. Префиксная / суффиксная схема.

Схема алфавитного кодирования называется префиксной/суффиксной, если ни одно кодовое слово не является префиксом/суффиксом другого слова.

46. Избыточность схемы алфавитного кодирования

Избыточность схемы - средняя длина ее кодового слова.

47. Формула подсчета избыточности схемы.

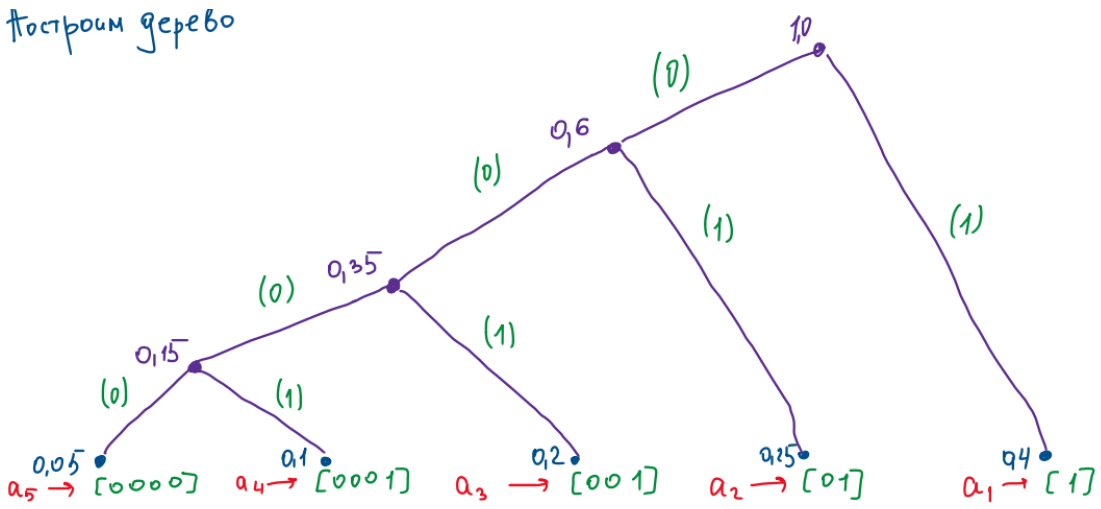
$$R = \sum_{i=1}^{r} p_i \, l_i$$

, где r - количество букв в исходном алфавите, q_1, q_2, …, q_r - частоты, l_1, l_2, …, l_r - длины кодовых слов.

48. Код Хаффмана.

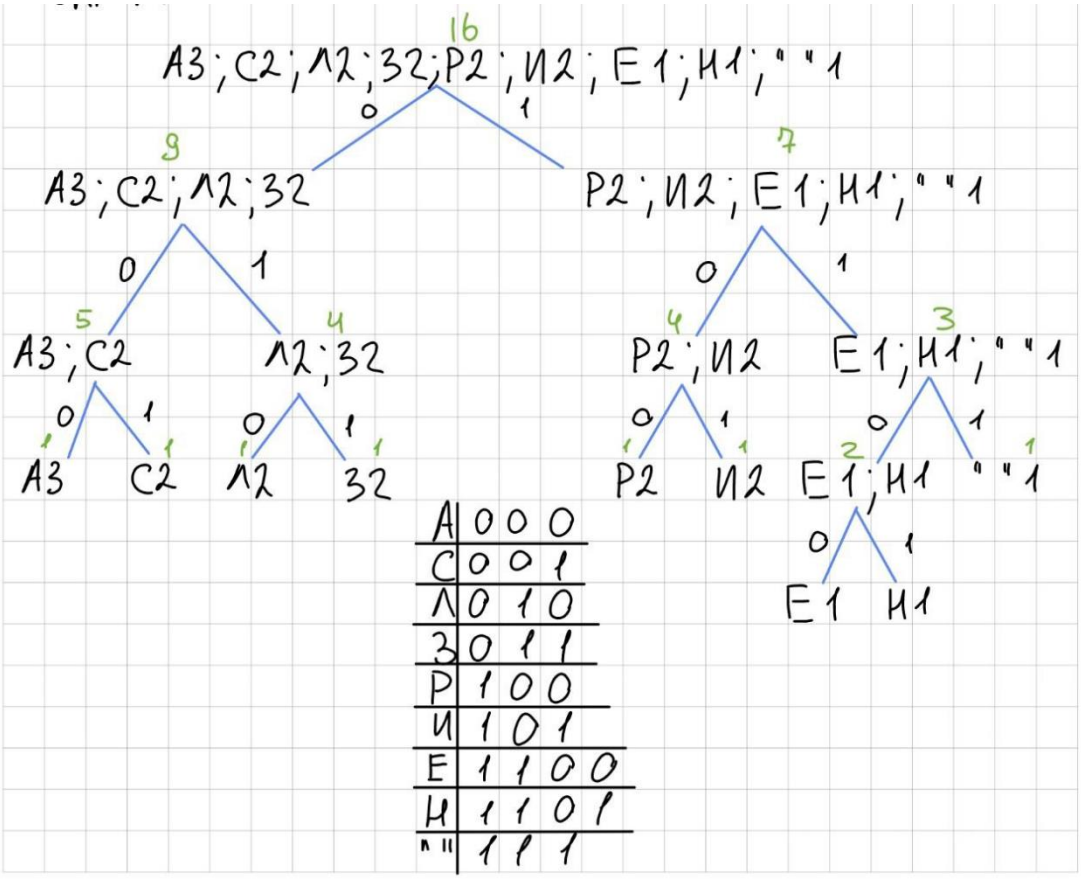
Имеем упорядоченный список пар (i; p_i) по возрастанию p_i, где i - символ, p_i - вероятность (или частота) символа.
Алгоритм кодирования (повторяем, пока список не будет пуст):
1. Строим дерево снизу вверх, берем первые две пары списка, они будут листьями, суммируем их вероятность - получаем родителя этих двух листьев, отображаем его на дереве.
2. Убираем эти 2 листа из списка пар, добавляем в список их родителя.
3. Пересортируем список.

После построения дерева отмечаем нули на «левых стрелках», единицы на «правых стрелках» и записываем код для каждого символа из 0 и 1, проходя от корня дерева до листьев.



49. Код Шеннона - Фано.

Имеем упорядоченный список пар (i; p_i) по убыванию p_i, где i - символ, p_i - вероятность (или частота) символа.
В корень дерева записываем весь список, далее разбиваем его на две (по возможности) равновероятные группы.
Делает это до тех пор, пока на листах дерева не будут списки, состоящие из одной пары.
После построения дерева отмечаем нули на «левых стрелках», единицы на «правых стрелках» и записываем код для каждого символа из 0 и 1, проходя от корня дерева до листьев.

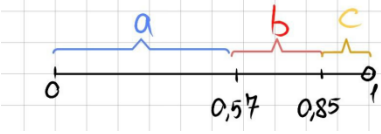


А	0 0 0
С	0 0 1
Л	0 1 0
З	0 1 1
Р	1 0 0
И	1 0 1
Е	1 1 0 0
Н	1 1 0 1
" "	1 1 1

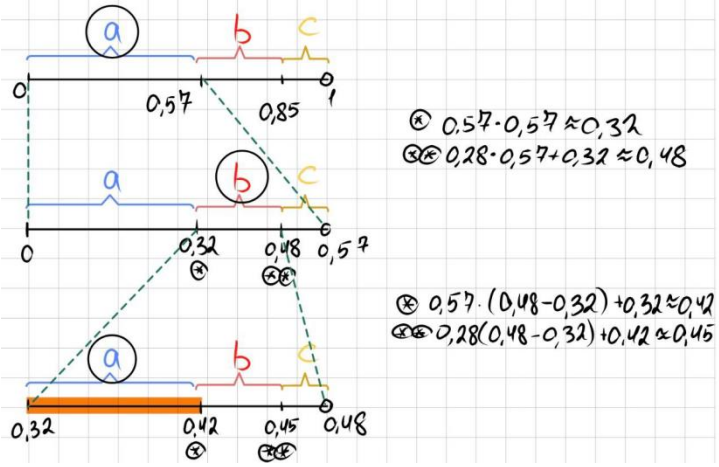
50. Арифметическое кодирование.

Имеем текст, который необходимо закодировать, и список вероятностей символов. Рассматривается полуинтервал [0; 1) на координатной прямой.
Алгоритм (закодируем сообщение aba):
1. Отмечаем на полуинтервале символы по порядку, согласно их вероятности:

Символ	Частота появления
a	0,571429
b	0,285714
c	0,142857



2. Считываем символ, рассматриваем соответствующий ему отрезок и делим его в том же соотношении (и так для всех символов):



3. Берем любое число с итогового (оранжевого) полуинтервала, например - 0.41, что и будет являтся результатом кодирования.

Для декодирования нужно знать: альфавит с частотами, само сообщение (0,41), количество символов в исходном сообщении. Также делим полуинтервал [0; 1) и смотрим, в какую из частей попадет 0,41. Так делаем n раз, записывая, какому символу соответствует текущий полуинтервал.

51. Расстояние Хэмминга.

Расстояние Хэмминга - количество позиций, на которое отличаются два закодированных сообщения одинаковой длины.

52. Код Хэмминга.

Контрольные биты расположены на степенях двойки, между ними - коды символов сообщения.

Для того, чтобы понять, за какие биты отвечает каждых контрольный бит необходимо понять очень простую закономерность:

- контрольный бит с номером **N** контролирует все последующие **N** бит через каждые **N** бит, начиная с позиции **N**.

Пример ниже

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	1	1	1	0	1	
X		X		X		X		X		X		X		X		X		X		X	1
	X	X			X	X			X	X			X	X			X	X			2
			X	X	X	X					X	X	X	X					X	X	4
							X	X	X	X	X	X	X	X							8
															X	X	X	X	X	X	16

Для заполнения контрольных битов складываем по модулю 2 все биты, за которые отвечает соответствующий контрольный.

53. Принцип поиска и исправление ошибок в коде Хэмминга.

Для исправления ошибки пересчитываем контрольные биты, если получили другое значение, значит есть ошибка. Складывая номера неверных контрольных битов можно получить позицию неверного бита.
Чтобы корректно обнаружить 2ую ошибку (без исправления), нужно использовать расширенный код Хэмминга (в начало добавляем контрольный бит, складывающий по модулю 2 все биты после него).

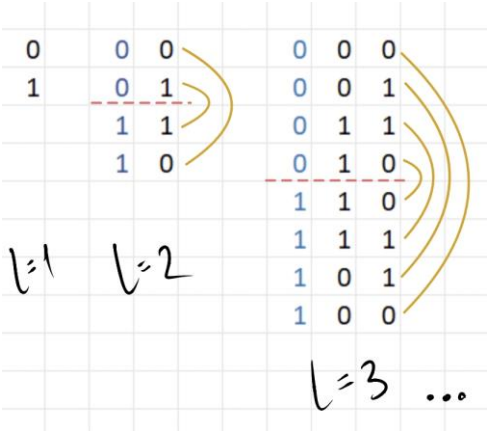
54. Код Грея.

Код Грея - двоичный код, в котором расстояние Хэмминга между соседними кодовыми словами равно 1.
Если первое и последние кодовые слова отличаются только в одном разряде, то такой код Грея будет циклическим.

Кодирование зеркального кода Грея:

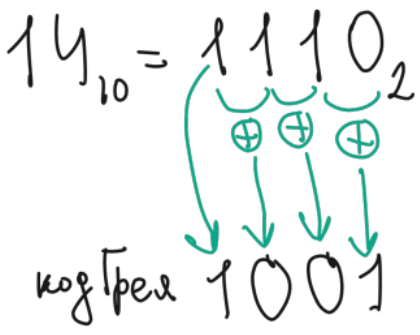
✧ *Способ 1*

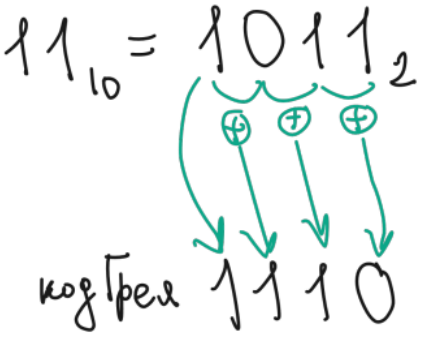
Строим код длины l = 1, затем отзеркаливаем (красный пунктир) и первой половине слов в начале приписываем ноль, второй - единицу. Так можно повторять бесконечно много раз.



✧ *Способ 2*

Идем по разрядам справа налево, выполняем XOR между текущим разрядом и разрядом слева. Последний разряд переписываем (XOR выполняется с незначащим нулем):





✧ *Способ 3*

Ищем i-ый код Грея по формуле $i \oplus \lfloor i/2 \rfloor$.
Пример для i = 11:

$$11 \oplus \lfloor 1/2 \rfloor = 11 \oplus 5 = 1011 \oplus 0101 = 1110$$

55. Циклический код.

Циклический код - блочный линейный код, обладающий свойством цикличности (любая циклическая перестановка кодового слова также является кодовым словом).
Блочный код - код, увеличивающий избыточность сообщения для декодирования с минимальным количеством ошибок.

Линейный код - тип блокового кода, в котором любая линейная комбинация кодовых слов также является кодовым словом этого кода. Линейная комбинация для двоичных кодов - сложение по модулю 2.

Несистематический циклический код - код, в котором нет явного разделения на информационные и контрольные биты.

Систематический циклический код - код, в котором есть явное разделение на информационные и контрольные биты.