ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики им. А.Н. Тихонова

Ефремов Виктор Васильевич, группа БИТ 203

Отчет по домашней работе 1
по дисциплине "Информатика"Тема: "Математические основы вычислительной техники"

Номер варианта: 6
Дата сдачи отчета: 18.10.2020

Москва 2020

1.1

Перевести числа в двоичное представление. Из него в восьмиричное и шестнадцатиричное. Сделать проверку.

$$a = 496.82$$

$$b = 100.635$$

$$c = 606.274$$

$$d = 747.33$$

$$e = 380.12$$

$$f = 552.806$$

Сделаем только первое число с комментариями и пояснениями, т.к. алгоритм одинаков для всех. Будем переводить отдельно целую часть, отдельно дробную.

Рассмотрим первое число $a = 496.82$.

Переведем его целую часть в двоичную систему счисления. Для этого будем делить число нацело на 2 и записывать остатки. Искомое двоичное представление - это остатки записанные в обратном порядке.

$$\begin{aligned}
496 &= 248 \cdot 2 + 0 \\
248 &= 124 \cdot 2 + 0 \\
124 &= 62 \cdot 2 + 0 \\
62 &= 31 \cdot 2 + 0 \\
31 &= 15 \cdot 2 + 1 \\
15 &= 7 \cdot 2 + 1 \\
7 &= 3 \cdot 2 + 1 \\
3 &= 1 \cdot 2 + 1 \\
1 &= 0 \cdot 2 + 1
\end{aligned}$$

$$496_{10} = 111110000_2$$

Такой способ записи, хотя и ясный, слишком громоздкий. Поэтому в дальнейшем будем просто записывать числа в две колонки: слева частное, справа остаток.

496	
248	0
124	0
62	0
31	0
15	1
7	1
3	1
1	1
0	1

Для перевода дробной части умножаем её на 2 и записываем целую часть результата (0 или 1). Повторяем до тех пор, пока дробная часть не станет нулем, или до достижения нужной точности. Целые части взятые в прямом порядке - двоичное представление дробной части.

Замечание про точность и знаки после точки. Чем мньше основание системы счисления, тем больше знаков нужно для той же точности. Если мы переводим из q-ичные числа в p-ичные, то на каждый знак после точки в исходном числе нужно $log_p q$ знаков в результате. Например для перевода из десятичной в двоичную на каждый знак десятичный нужно $log_2 10 \approx 3.3219$ двоичных знаков. Таким образом для двух знаков после точки в десятичной записи нужно 7 (округляем в большую сторону) в двоичной, а для трех - 10.

$0.82 \cdot 2 = 0.64 + 1$

$0.64 \cdot 2 = 0.28 + 1$

$0.28 \cdot 2 = 0.56 + 0$

$0.56 \cdot 2 = 0.12 + 1$

$0.12 \cdot 2 = 0.24 + 0$

$0.24 \cdot 2 = 0.48 + 0$

$0.48 \cdot 2 = 0.96 + 0$

$$0.82_{10} \approx 1101000_2$$

Эта запись также громоздка, поэтому будем записывать только последовательные результаты умножения дробной части на 2:

$0.82 - 1.64 - 1.28 - 0.56 - 1.12 - 0.24 - 0.48 - 0.96$

Итого

$$496.82_{10} \approx 111110000.1101000_2$$

Стоит помнить, что это неточное представление. Оно лишь имеет сходную точность (примерно до сотой).

Переведем теперь число из двоичной в восьмиричную и шестнадцатиричную систеы счисления. Для этого разобьем цифры по 3 (т.к. $8 = 2^3$) и 4 ($16 = 2^4$) начиная от разделителя-точки и дописывая незначащие нули, если цифр не хватает.

$111\,110\,000.110\,100\,000_2 = 760.64_8$

$0001\,1111\,0000.1101\,0000_2 = 1F0.D0_{16}$

Проверку сделаем просто любым калькулятором/конвертером.

Например https://binary2hex.ru/numberconverter.html

$b = 100.635$

100	
50	0
25	0
12	1
6	0
3	0
1	1
0	1

$100_{10} = 1100100_2$

$0.635 - 1.270 - 0.540 - 1.080 - 0.160 - 0.320 - 0.640 - 1.280 - 0.560 - 1.120 - 0.240$
$0.635_{10} \approx 1010001010_2$

$100.635_{10} \approx 1100100.1010001010_2$
$001\,100\,100.101\,000\,101\,000_2 = 144.5050_8$
$0110\,0100.1010\,0010\,1000_2 = 64.A28_{16}$

$c = 606.274$

606	
303	0
151	1
75	1
37	1
18	1
9	0
4	1
2	0
1	0
0	1

$606_{10} = 1001011110_2$

$0.274 - 0.548 - 1.096 - 0.192 - 0.384 - 0.768 - 1.536 - 1.072 - 0.144 - 0.288 - 0.576$
$0.274_{10} \approx 0.0100011000_2$

$606.274_{10} \approx 1001011110.0100011000_2$
$001\,001\,011\,110.010\,001\,100\,000_2 = 1136.2140_8$
$0010\,0101\,1110.0100\,0110\,0000_2 = 25E.460_{16}$

$d = 747.33$

747	
373	1
186	1
93	0
46	1
23	0
11	1
5	1
2	1
1	0
0	1

$747_{10} = 1011101011_2$

$0.33 - 0.66 - 1.32 - 0.64 - 1.28 - 0.56 - 1.12 - 0.24$
$0.33_{10} \approx 0.0101010_2$

$747.33_{10} \approx 1011101011.0101010_2$
$001\,011\,101\,011.010\,101\,000_2 = 1353.25_8$
$0010\,1110\,1011.0101\,0100_2 = 2EB.54_{16}$

$e = 380.12$

380	
190	0
95	0
47	1
23	1
11	1
5	1
2	1
1	0
0	1

$380_{10} = 101111100_2$

$0.12 - 0.24 - 0.48 - 0.96 - 1.92 - 1.84 - 1.68 - 1.36$
$0.12_{10} \approx 0001111_2$

$380.12_{10} \approx 101111100.0001111_2$
$101\,111\,100.000\,111\,100_2 = 574.074_8$
$0001\,0111\,1100.0001\,1110_2 = 17C.1E_{16}$

$f = 552.806$

552	
276	0
138	0
69	0
34	1
17	0
8	1
4	0
2	0
1	0
0	1

$552_{10} = 1000101000_2$

$0.806 - 1.612 - 1.224 - 0.448 - 0.896 - 1.792 - 1.584 - 1.168 - 0.336 - 0.672 - 1.344$
$0.806_{10} \approx 0.1100111001_2$

$552.806_{10} \approx 1000101000.1100111001_2$
$001\,000\,101\,000.110\,011\,100\,100_2 = 1050.6344_8$
$0010\,0010\,1000.1100\,1110\,0100_2 = 228.CE8_{16}$

1.2

Выполнить умножение и деление двоичных чисел. Сделать проверку.

$X_1 = a \cdot b = 496.82 \cdot 100.635 \approx 111110000.1101000 \cdot 1100100.1010001010$
Для простоты умножения отбросим незначащие нули, а также точку-разделитель.

```
                    1111100001101 000
                 1100100101000101 0
--------------------------------------
                    1111100001101
                  1111100001101
              1111100001101
            1111100001101
         1111100001101
      1111100001101
     1111100001101
--------------------------------------
    1100001101001100100111000001 0000

    1100001101001100.10011100000010000
```

$X_1 \approx 1100001101001100.10011100000010000$
Проверка:
$X_1 - a \cdot b \approx 49996.6094970703125_{10} - 496.82_{10} \cdot 100.635_{10} \approx -0.87120292968$
Стоит отметить что погрешность X_1 на уровне единиц.

$X_3 = c \cdot d = 606.274 \cdot 747.33 \approx 1001011110.0100011000 \cdot 1011101011.0101010$

```
                 10010111100100011 000
                  1011101011010101 0
--------------------------------------
                 10010111100100011
               10010111100100011
             10010111100100011
           10010111100100011
          10010111100100011
        10010111100100011
      10010111100100011
     10010111100100011
    10010111100100011
  10010111100100011
--------------------------------------
  11011101001110111010011000011111 0000

  1101110100111011101.00110000111110000
```

$X_3 \approx 1101110100111011101.00110000111110000$
Проверка:
$X_3 - c \cdot d \approx 453085.1912841796875_{10} - 606.274_{10} \cdot 747.33_{10} \approx -1.55713582033$

1.3

Перевести данное число в десятичное из а) беззнакового б) дополнительного кода

$A88D_{16} = 1010\,1000\,1000\,1101_2$

а) Число без знака, поэтому просто складываем степени двойки:

$$2^{15} + 2^{13} + 2^{11} + 2^7 + 2^3 + 2^2 + 2^0 = 32768 + 8192 + 2048 + 128 + 8 + 4 + 1 = 43149$$

Ответ: 43149.

б) Число со знаком, дополнительный код. Самый старший (левый) бит - это знак, все остальное это модуль. При этом нужно инвертировать биты и прибавить единицу, т.к. у нас отрицательное число. Инверсия $010\,1000\,1000\,1101_2$ - это $101\,0111\,0111\,0010_2$, прибавляем единицу $101\,0111\,0111\,0011_2$, переводим в десятичную систему счисления:

$$2^{14} + 2^{12} + 2^{10} + 2^9 + 2^8 + 2^6 + 2^5 + 2^4 + 2^1 + 2^0 = 22387$$

Ответ: -22387.

1.4

Записать числа в прямом, обратном и дополнительном кодах. Выполнить сложение. Отметить переполнения.

Запишем все числа для удобства в таблицу.

Во всех трех кодах первый/старший бит - это знак. Оставшиеся биты представляют модуль числа.

Стоит помнить, что все три кода совпадают на положительных числах.

Для отрицательных чисел и нуля алгоритм следующий.

Чтобы получить прямой код просто переводим число из десятичного в двоичное. Добавляем незначащих ведущих нулей при необходимости. Не забываем про единицу знака в старшем бите.

Модуль числа в обратном коде получается из прямого инверсией бит.

Дополнительный код получается из обратного добавлением единицы.

	десятичное	прямой	обратный	дополнительный
a	496	00000001 11110000	00000001 11110000	00000001 11110000
b	100	00000000 01100100	00000000 01100100	00000000 01100100
c	606	00000010 01011110	00000010 01011110	00000010 01011110
d	747	00000010 11101011	00000010 11101011	00000010 11101011
e	-380	10000001 01111100	11111110 10000011	11111110 10000100
f	-552	10000010 00101000	11111101 11010111	11111101 11011000

Выполним сложение.

$X_5 = a + b$

Сложим их в прямом коде. Т.к. оба положительны, никаких дополнительных действий делать не нужно.

```
    0000 0001 1111 0000
+   0000 0000 0110 0100
-----------------------
    0000 0010 0101 0100
```

$X_5 = 00000010\,01010100$

$X_6 = c + e$

Эти два сложим в обратном коде. Т.к. у нас в сумме появляется единица, которая не влезает в 16 бит, мы переносим её направо и складываем с ответом.

```
    0000 0010 0101 1110
+   1111 1110 1000 0011
-----------------------
  1 0000 0000 1110 0001
+                     1
-----------------------
    0000 0000 1110 0010
```

$X_6 = 00000000\,11100010$

$X_7 = d + f$

А это сложим в дополнительном коде. В дополнительном коде вылезающая за границы единица просто отбрасывается.

```
    0000 0010 1110 1011
+   1111 1101 1101 1000
-----------------------
  1 0000 0000 1100 0011
-----------------------
    0000 0000 1100 0011
```

$X_7 = 00000000\,11000011$

1.5

Записать числа в float из стандарта IEEE 754. Произвести сложение. Сделать проверку.

Биты в флоате распределены следующим образом: 1 для знака, 8 для порядка/экспоненты, 23 для мантиссы.

Для представления числа в IEEE 754 формате удобно предварительно записать его в виде

$$(-1)^s \cdot 2^e \cdot m$$

Где s - знак, e - экспонента, m - мантисса.

Стоит помнить, что экспонента записывается в коде со сдвигом 127. Мантисса не меньше единицы и меньше двух, т.е. $1 \leq m < 2$. Из-за этого целая часть мантиссы всегда равна 1 и не записывается.

Ещё один неочевидный момент - это округление мантиссы. По-умолчанию используется "round to nearest, ties to even"(до ближайшего; до четного, если ближайшие равноудалены).

Рассмотрим такое округление на нескольких примерах. Все числа в примерах двоичные.

Пусть мы хотим округлить 100.1 до целого. Кандидатов быть ответом два: 100 и 101. Но 100.1 в *точности* по середене между 100 и 101, поэтому мы округляем к четному, т.е. числу с нулем на конце. Ответ 100.

Округлим число 101.1000000000010 до целого. Кандидатов опять же два: 101 и 110. 101.1000000000010 ближе к 110. Действительно, $110 - 101.1000000000010 = 0.0111111111110$, $101 - 101.1000000000010 = -0.1000000000010$. Первое расстояние меньше $0.1_2 = 0.5_{10}$, а второе больше. Поэтому ответ - 110.

Запишем число $a = 496.82 \approx 111110000.1101000$.

Найденое в 1.1 представление недостаточно точное. Нам нужно больше знаков после точки. Посчитаем.

$0.82 - 1.64 - 1.28 - 0.56 - 1.12 - 0.24 - 0.48 - 0.96 - 1.92 - 1.84 - 1.68 - 1.36 - 0.72 - 1.44 - 0.88 - 1.76 - 1.52 - 1.04 - 0.08 - 0.16 - 0.32 - 0.64$

$a \approx 111110000.11010001111010111000 0_2 = (-1)^0 \cdot 1.11110000110100011110101\,110000_2 \cdot 10_2^{1000_2}$

Знак $s = 0$, т.к. число положительное.

Экспонента равна $8_{10} = 1000_2$, но т.к. мы её записываем в коде со сдвигом, к ней нужно прибавить $127_{10} = 111\,1111_2$. Поэтому $e = 10000111$.

Посмотрим на мантиссу. Нам нужно 23 двоичных знака после точки. Но просто отрезать хвост нельзя, т.к. по стандарту мантисса округляется "to nearest, ties to even". Поэтому мантисса будет равна $m = 11110000110100011110101$.

Таким образом число a записывается как

$$a_{float} = 01000011111110000110100011110110$$

$b = 100.635$

$0.635-1.270-0.540-1.080-0.160-0.320-0.640-1.280-0.560-1.120-0.240-0.480-0.960-1.920-1.840-1.680-1.360-0.720-1.440-0.880-1.760-1.520-1.040-0.080$

$b \approx 1100100.101000101000111101011110_2 = 1.100100101000101000111101011110 \cdot 10^{110}$

$s = 0$

$e = 10000101$

$m = 10010010100010100011111$

$$b_{float} = 01000010110010010100010100011111$$

$c = 606.274$

$0.274-0.548-1.096-0.192-0.384-0.768-1.536-1.072-0.144-0.288-0.576-1.152-0.304-0.608-1.216-0.432-0.864-1.728-1.456-0.912-1.824-1.648-1.296-0.592$

$c \approx 1001011110.0100011000100100110110 = 1.0010111100100011000100100110110 \cdot 10^{1001}$

$s = 0$

$e = 10001000$

$m = 00101111001000110001001$

$$c_{float} = 01000100000101111001000110001001$$

$d = 747.33$

$0.33-0.66-1.32-0.64-1.28-0.56-1.12-0.24-0.48-0.96-1.92-1.84-1.68-1.36-0.72-1.44-0.88-1.76-1.52-1.04-0.08-0.16$

$d \approx 1011101011.0101010001111010111100 = 1.0111010110101010001111010111100 \cdot 10^{1001}$

$s = 0$

$e = 10001000$

$m = 01110101101010100011111$

$$d_{float} = 01000100001110101101010100011111$$

$e = 380.12$

$0.12-0.24-0.48-0.96-1.92-1.84-1.68-1.36-0.72-1.44-0.88-1.76-1.52-1.04-0.08-0.16-0.32-0.64-1.28-0.56-1.12-0.24n$

$e \approx 101111100.0001111010111000010100 = 1.0111110000011110101110000101 0 \cdot 10^{1000}$

$s = 0$

$e = 10000111$

$m = 01111100000111101011100$

$$e_{float} = 01000011101111100000111101011100$$

$f = 552.806$

$0.806-1.612-1.224-0.448-0.896-1.792-1.584-1.168-0.336-0.672-1.344-0.688-1.376-0.752-1.504-1.008-0.016-0.032-0.064-0.128-0.256-0.512-1.024-0.048$

$f \approx 1000101000.110011100101011000000010 = 1.000101000110011100101011000000010 \cdot 10^{1001}$

$s = 0$

$e = 10001000$

$m = 00010100011001110010110$

$$f_{float} = 01000100000010100011001110010110$$

Сложим числа.

$X_8 = a + c = 496.82 + 606.274$.

$a_{float} = 1.11110000110100011110110 \cdot 10^{1000}$

$c_{float} = 1.00101111001000110001001 \cdot 10^{1001}$

Для сложения чисел нужно приравнять их экспоненты, сложить мантиссы и нормализовать результат.

В нашем случае приведем оба числа к экспоненте 1001 и сложим.

```
    0.111110000110100011110110
+   1.00101111001000110001001
------------------------------
   10.001001111000110000001000
```

$10.001001111000110000001000 \cdot 10^{1001} = 1.0001001111000110000001000 \cdot 10^{1010}$

$X_8 = 01000100100010011110001100000010 = 1103.093994140625$

Проверка:

$496.82 + 606.274 - 1103.093994140625 = 0.00000585937$

$X_9 = b + e = 100.635 + 380.12$

$b_{float} = 1.10010010100010100011111 \cdot 10^{110}$

$e_{float} = 1.01111100000111101011100 \cdot 10^{1000}$

Приведем к экспоненте 1000:

```
    0.0110010010100010100011111
+   1.01111100000111101011100
------------------------------
    1.1110000011000001010001111
```

$X_9 = 01000011111100000110000010100100 = 480.7550048828125$

Проверка:

$100.635 + 380.12 - 480.7550048828125 = -0.00000488281$

$X_{10} = d + f = 747.33 + 552.806$

$d_{float} = 1.01110101101010100011111 \cdot 10^{1001}$

$f_{float} = 1.00010100011001110010110 \cdot 10^{1001}$

Здесь экспоненты уже совпадают.

```
    1.01110101101010100011111
+   1.00010100011001110010110
------------------------------
   10.10001010000100010110101
```

$10.10001010000100010110101 \cdot 10^{1001} = 1.010001010000100010110101 \cdot 10^{1010}$

$X_{10} = 01000100101000101000010001011010$

Проверка:

$747.33 + 552.806 - 1300.135986328125 = 0.00001367187$

2.1

$f_1 = \{0, 1, 4, 6, 7, 8, 9, 10, 11, 12, 13, 15\}$

Запишем таблицу истинности для f:

abcd	f
0000	1
0001	1
0010	0
0011	0
0100	1
0101	0
0110	1
0111	1
1000	1
1001	1
1010	1
1011	1
1100	1
1101	1
1110	0
1111	1

2.1.1

Записать СДНФ и СКНФ.

Как мы помним, ДНФ - это сумма произведений (или дизъюнкция конъюнкций). СДНФ (совершенная ДНФ) содержит только те термы, на которых функция равна 1. Поэтому СДНФ это:

$$\overline{a}\overline{b}\overline{c}\overline{d} + \overline{a}\overline{b}\overline{c}d + \overline{a}b\overline{c}\overline{d} + \overline{a}bc\overline{d} + \overline{a}bcd + a\overline{b}\overline{c}\overline{d} + a\overline{b}\overline{c}d + a\overline{b}c\overline{d} + a\overline{b}cd + ab\overline{c}\overline{d} + ab\overline{c}d + abcd$$

Аналогично, КНФ - это произведение сумм (или конъюнкция дизъюнкций). В СКНФ используются термы на которых функция равна 0. СКНФ:

$$(a + b + \overline{c} + d)(a + b + \overline{c} + \overline{d})(a + \overline{b} + c + \overline{d})(\overline{a} + \overline{b} + \overline{c} + d)$$

2.1.2

Минимизировать функцию с помощью метода Квайна для СДНФ.

Выпишем все наборы, на которых $f = 1$ и пронумеруем их.

Проведем операцию *склейки*. Для этого ищем строки таблицы различающиеся только в одной переменной и склеиваем их. Записываем результат в соседнюю таблицу, а сами строки помечаем, например серым фоном.

Например термы $\overline{a}\overline{b}\overline{c}\overline{d}$ и $\overline{a}\overline{b}\overline{c}d$ различаются только в переменной d. Поэтому их можно склеить.

Делаем это для всех возможных пар.

Повторяем процесс для второй и последующих таблиц.

1	$\overline{a}\overline{b}\overline{c}\overline{d}$
2	$\overline{a}\overline{b}\overline{c}d$
3	$\overline{a}b\overline{c}\overline{d}$
4	$\overline{a}bc\overline{d}$
5	$\overline{a}bcd$
6	$a\overline{b}\overline{c}\overline{d}$
7	$a\overline{b}\overline{c}d$
8	$a\overline{b}c\overline{d}$
9	$a\overline{b}cd$
10	$ab\overline{c}\overline{d}$
11	$ab\overline{c}d$
12	$abcd$

1 - 2	$\overline{a}\overline{b}\overline{c}$
1 - 3	$\overline{a}\overline{c}\overline{d}$
1 - 6	$\overline{b}\overline{c}\overline{d}$
2 - 7	$\overline{b}\overline{c}d$
3 - 4	$\overline{a}b\overline{d}$
3 - 10	$b\overline{c}\overline{d}$
4 - 5	$\overline{a}bc$
5 - 12	bcd
6 - 7	$a\overline{b}\overline{c}$
6 - 8	$a\overline{bd}$
6 - 10	$a\overline{c}\overline{d}$
7 - 9	$a\overline{b}d$
7 - 11	$a\overline{c}d$
8 - 9	$a\overline{b}c$
9 - 12	acd
10 - 11	$ab\overline{c}$
11 - 12	abd

1 - 2 - 6 - 7	$\overline{b}\overline{c}$
1 - 3 - 6 - 10	$\overline{c}\overline{d}$
1 - 6 - 2 - 7	$\overline{b}\overline{c}$
1 - 6 - 3 - 10	$\overline{c}\overline{d}$
6 - 7 - 8 - 9	$a\overline{b}$
6 - 7 - 10 - 11	$a\overline{c}$
6 - 8 - 7 - 9	$a\overline{b}$
6 - 10 - 7 - 11	$a\overline{c}$
7 - 9 - 11 - 12	ad
7 - 11 - 9 - 12	ad

Посмотрим на не выделенные строки в таблицах выше. В них записаны термы, которые ни с чем не склеились. Это *простые импликанты*. Минимальная ДНФ - это сумма некотоорых (возможно всех) простых импликант.

Чтобы выяснить какие импликанты нужно брать в миинимальную форму заполним так называемую импликантную матрицу. Протые импликанты записываются по строкам, слагаемые из СДНФ по столбцам.

Если импликанта поглощает терм СДНФ (т.е. целиком входит в него), то помечаем пересечение соответствующих строки и столбца (например галочкой).

Найдем галочки, которые единственные в своем столбце. Выделим их.

Строки, в которых есть выделенные галочки - ядро. Все импликанты составляющие ядро обязательно будут в минимальной форме. В таблице ядро выделено зеленым.

Видно, что ядром остались не покрыты 6 столбцов. Если взять в дополнение к ядру строки 3, 5 и 7, то получим минимальную ДНФ.

	$\overline{a}\overline{b}\overline{c}\overline{d}$	$\overline{a}\overline{b}\overline{c}d$	$\overline{a}b\overline{c}\overline{d}$	$\overline{a}bc\overline{d}$	$\overline{a}bcd$	$a\overline{b}\overline{c}\overline{d}$	$a\overline{b}\overline{c}d$	$a\overline{b}c\overline{d}$	$a\overline{b}cd$	$ab\overline{c}\overline{d}$	$ab\overline{c}d$	$abcd$
$a\overline{b}$						✓	✓	✓	✓			
$a\overline{c}$						✓	✓			✓	✓	
ad							✓		✓		✓	✓
$\overline{b}\overline{c}$	✓	✓				✓	✓					
$\overline{c}\overline{d}$	✓		✓			✓				✓		
$\overline{a}b\overline{d}$			✓	✓								
$\overline{a}bc$				✓	✓							
bcd					✓							✓

Таким образом в результате минимизации получается следующая ДНФ:

$$a\overline{b} + \overline{b}\overline{c} + ad + \overline{c}\overline{d} + \overline{a}bc$$

Чтобы убедиться, что это действительно МДНФ, нужно перебрать остальные варианты и проверить, что в них больше вхождений переменных.

2.1.3

Минимизировать функцию с помощью диаграммы Вейча для СДНФ.

Заполним диаграмму Вейча в соответствии с таблицей истинности. Как мы помним, диаграмма заполняется зигзагами, начиная с правого нижнего угла.

b

a

1	1	1	1
0	1	1	1
1	1	0	0
1	0	1	1

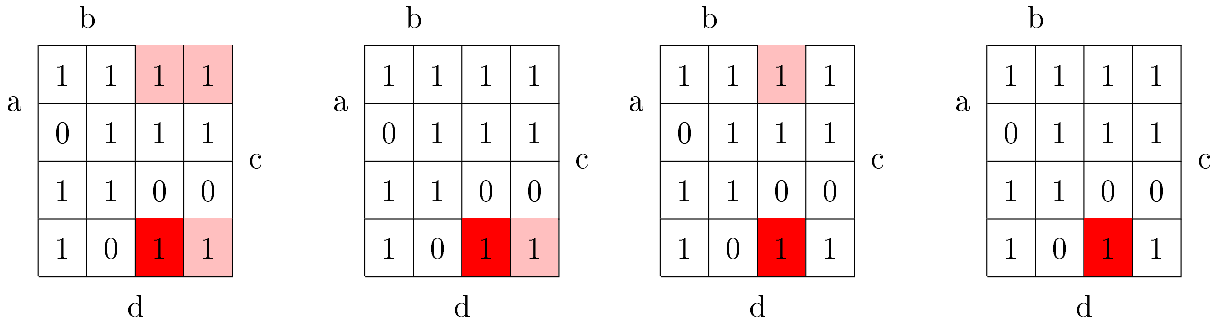
c

d

Посмотрим на элемент $\overline{a}\overline{b}\overline{c}d$. Его можно накрыть четыремя вариантами, которые изображены ниже.

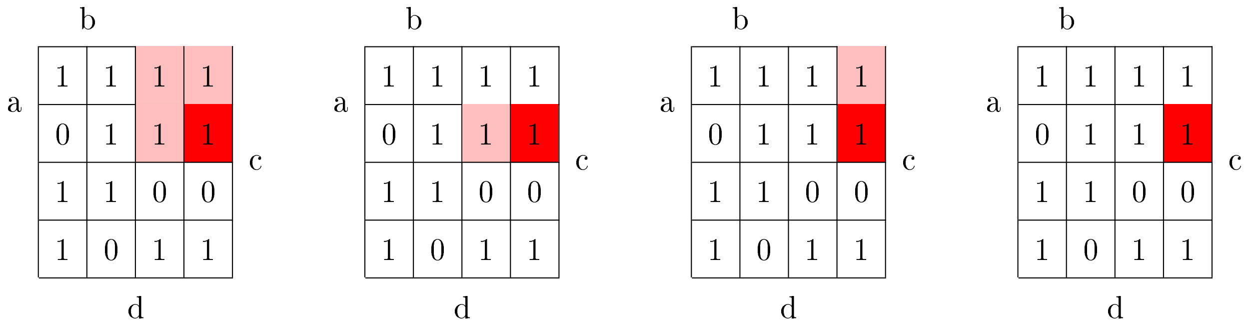
Заметим, что покрытие из четырех клеток включает в себя три других варианта. Это значит, что эта группа попадает в ядро.

Ей соответствует терм $\overline{b}\overline{c}$.



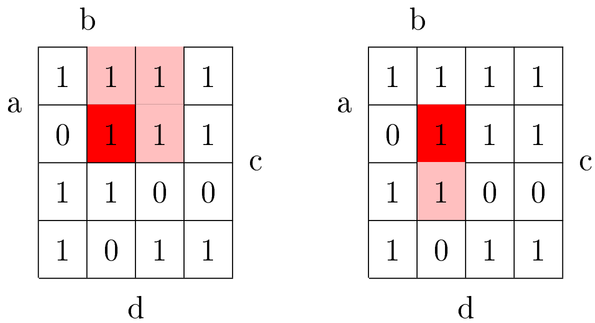
$a\bar{b}c\bar{d}$ абсолютно аналогичен. Покрытие из четырех клеток включат в себя все остальные.

Этой группе соответствует терм $a\bar{b}$.



Заметим, что описанные две группы и есть ядро. Это следует из того, что для любой из оставшихся единиц покрытие максимального размера не содержит в себе всех меньших.

Для иллюстрации посмотрим на клетку $abcd$ и два её покрытия, изображенных ниже. Большее покрытие, хоть и является единственной группой размера четыре, не входит в ядро, так как оно не содержит в себе изображенную группу размера два.



Посмотрим теперь на шесть оставшихся единиц, которые не покрыты ядром. Можно заметить, что покрытие изображенное ниже оптимально. Ему соответствует сумма $ad + \bar{c}\bar{d} + \bar{a}bc$. В ней семь вхождений переменных. Любое другое покрытие будет иметь восемь или более.

b

a

1	1	1	1
0	1	1	1
1	1	0	0
1	0	1	1

c

d

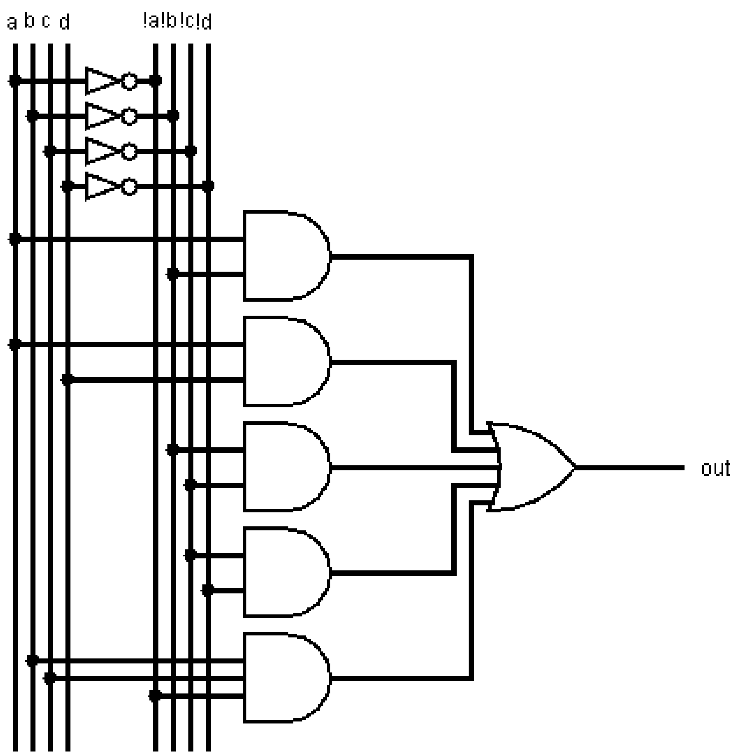
Поэтому минимальной ДНФ будет

$$a\overline{b} + ad + \overline{b}\overline{c} + \overline{c}\overline{d} + \overline{a}bc$$

2.1.4

Построить логическую схему.

Схему рисовал в программе Logisim



2.2

Задача во многом аналогична 2.1, поэтому комментарии только об отличающихся частях решения.

$f_1 = \{0, 1, 2, 3, 4, 5, 8, 9, 12, 13, 14\}$

Запишем таблицу истинности для f:

$abcd$	f
0000	1
0001	1
0010	1
0011	1
0100	1
0101	1
0110	0
0111	0
1000	1
1001	1
1010	0
1011	0
1100	1
1101	1
1110	1
1111	0

2.2.1

Записать СДНФ и СКНФ.

СДНФ:

$$\overline{a}\overline{b}\overline{c}\overline{d} + \overline{a}\overline{b}\overline{c}d + \overline{a}\overline{b}c\overline{d} + \overline{a}\overline{b}cd + \overline{a}b\overline{c}\overline{d} + \overline{a}b\overline{c}d + a\overline{b}\overline{c}\overline{d} + a\overline{b}\overline{c}d + ab\overline{c}\overline{d} + ab\overline{c}d + abc\overline{d}$$

СКНФ:

$$(a + \overline{b} + \overline{c} + d)(a + \overline{b} + \overline{c} + \overline{d})(\overline{a} + b + \overline{c} + d)(\overline{a} + b + \overline{c} + \overline{d})(\overline{a} + \overline{b} + \overline{c} + \overline{d})$$

2.2.2

Минимизировать функцию с помощью метода Квайна-МакКласки для СКНФ.

Алгоритм Квайна-МакКласки - это вариация метода Квайна. Основных изменений всего два. Во-первых термы записываются последовательностью 0 и 1, а не переменных и отрицаний. Во-вторых на этапе склейки термы группируются по количеству единиц. Это убирает некоторые бесполезные сравнения.

Выпишем все наборы, на которых $f = 0$ и пронумеруем их.

При этом используем цифровое представление вместо буквенного. Например вместо $a + b + c + \overline{d}$ пишем 0001.

Также сгруппируем наборы по количеству единиц.

Проведем операцию *склейки*. Для этого ищем строки таблицы различающиеся только в одной переменной и склеиваем их. Записываем результат в соседнюю таблицу, а сами строки помечаем, например серым фоном.

Аналогично методу Квайна склеиваем строки различающиеся только в одной переменной. Легко заметить, что склеить можно только строки из соседних групп. Т.е. те, у которых количество единиц различается ровно на один.

Вместо переменной по которой склеили ставим x, т.к. она для нас теперь не важна.

1	0110
2	1010
3	0111
4	1011
5	1111

3 - 5	x111
4 - 5	1x11
1 - 3	011x
2 - 4	101x

Заполняем импликантная матрицу, расставляем галочки, обводим их, находим ядро и дополнение, записываем минимальнуу КНФ. Все так же, как методе Квайна. Не меняется ничего.

	0110	1010	0111	1011	1111
x111			✓		✓
1x11				✓	✓
011x	✓		✓		
101x		✓		✓	

Ядро покрыло все, кроме последнего столбца. В нем есть две галолчки, поэтому можно взять либо первую, либо вторую строки.

Таким образом в результате минимизации получается две тупиковых КНФ:

$$(a + \overline{b} + \overline{c})(\overline{a} + b + \overline{c})(\overline{a} + \overline{c} + \overline{d})$$

$$(a + \overline{b} + \overline{c})(\overline{a} + b + \overline{c})(\overline{b} + \overline{c} + \overline{d})$$

2.2.3

Минимизировать функцию с помощью карт Карно для СКНФ.

Карты Карно подобны диаграммам Вейча. Главное отличие - расположение переменных, и соответственно порядок строк/столбцов.

Т.к. мы минимизирум КНФ, то группировать будем нули.

$ab \backslash cd$	00	01	11	10
00	1	1	1	1
01	1	1	0	0
11	1	1	0	1
10	1	1	0	0

Легко заметить, что две группы отмеченные ниже входят в ядро.

$ab \backslash cd$	00	01	11	10
00	1	1	1	1
01	1	1	0	0
11	1	1	0	1
10	1	1	0	0

Соответствующие им множители в КНФ - это $(a + \overline{b} + \overline{c})$ и $(\overline{a} + b + \overline{c})$.

Оставшийся ноль можно покрыть двумя равнозначными способами:

$ab \backslash cd$	00	01	11	10
00	1	1	1	1
01	1	1	0	0
11	1	1	0	1
10	1	1	0	0

$ab \backslash cd$	00	01	11	10
00	1	1	1	1
01	1	1	0	0
11	1	1	0	1
10	1	1	0	0

Соответствующие множители: $(\overline{a} + \overline{c} + \overline{d})$ и $(\overline{b} + \overline{c} + \overline{d})$.

Поэтому в результате минимизации получатся две тупиковых КНФ:

$$(a + \overline{b} + \overline{c})(\overline{a} + b + \overline{c})(\overline{a} + \overline{c} + \overline{d})$$
$$(a + \overline{b} + \overline{c})(\overline{a} + b + \overline{c})(\overline{b} + \overline{c} + \overline{d})$$

2.2.4

Построить логическую схему.

Схему рисовал в программе Logisim