Конспект стрима №2

Logika_2

Задача № 1 (1607)

(№ 1607) Логическая функция F задаётся выражением (¬x ∧ y ∧ z) ∨ (¬x ∧ ¬z).

?	?	?	F
0	0	0	1
1	0	0	1
1	1	0	1

На рисунке приведён фрагмент таблицы истинности функции F, содержащий все наборы аргументов, при которых функция F истинна. Определите, какому столбцу таблицы истинности функции F соответствует каждая из переменных x, y, z.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=2m

Решение

Для решения этой и подобных задач будем использовать программу, которая подставляет переменные x, y, z в исходную таблицу в разном порядке, считает значение функции для каждой строки и сверяет с эталоном. Очевидно, что только в одном случае, только при одной расстановке переменных значения совпадут с теми, которые пропущены в таблице. Эта расстановка и будет результатом работы программы и ответом на вопрос задачи.

Последовательно разберём написание кода программы.

Т.к. в программе будет осуществляться перебор перестановок переменных будем использовать библиотеку функций для работы с итерируемыми объектами Itertools.

```
from itertools import *
```

Для удобства вычисления заданной логической функции, создадим отдельную функцию f(x, y, z), которая будет принимать внутрь себя значение переменных x, y, z, и возвращать значение заданного логического выражения.

```
from itertools import *


def f(x,y,z):
```

```
    return (not x and y and z) or (not x and not z)
```

Следующим шагом воссоздадим в программе таблицу, заданную в условии задачи. Для этого создадим список table, который будет содержать в себе три кортежа, или списка, соответствующих строкам таблицы, каждый список – значения переменных в одной строке, без значений функции.

```
from itertools import *


def f(x,y,z):
    return (not x and y and z) or (not x and not z)


table = [(0,0,0), (1,0,0), (1,1,0)]
```

Далее, переберём разные варианты перестановок x, y, z. Переменная p будет являться перестановкой букв.

```
from itertools import *

for p in permutations('xyz'):

        print(p)
```

Результат выполнения этих строк выглядит следующим образом:

```
('x', 'y', 'z')
('x', 'z', 'y')
('y', 'x', 'z')
('y', 'z', 'x')
('z', 'x', 'y')
('z', 'y', 'x')
```

Программа вывела все шесть различных перестановок.

Выполнив эти действия, мы перенесли информацию из задания в программу, записав ее в виде функции и в виде списка списков.

Следующий шаг – основной для решения задачи. Нам необходимо переменные, в определенном условием задачи расположении в строках, подставить в функцию в том же порядке и выяснить значение выражения.

Это реализуется в условии, в котором считается значение логического выражения для переменных в каждой строке таблицы. Если это значение оказывается равным значению функции для данной строки (в данной задаче это f=[1,1,1]), то найден тот порядок букв, при котором это условие выполняется.

```
from itertools import *
```

```
def f(x,y,z):
    return (not x and y and z) or (not x and not z)

table = [(0,0,0), (1,0,0), (1,1,0)]

for p in permutations('xyz'):
    if [f(**dict(zip(p,r))) for r in table] == [1,1,1]:
        print(p)
```

Таким образом мы рассматриваем разный порядок переменных, для каждого из них вычисляем значение строки и сравниваем со значениями f , заданными в таблице.

Результат работы программы - порядок букв для столбцов таблицы в задаче:

```
('y', 'z', 'x')
```

Разберем принцип работы этой строки в программы.

```
if [f(**dict(zip(p,r))) for r in table] == [1,1,1]:
```

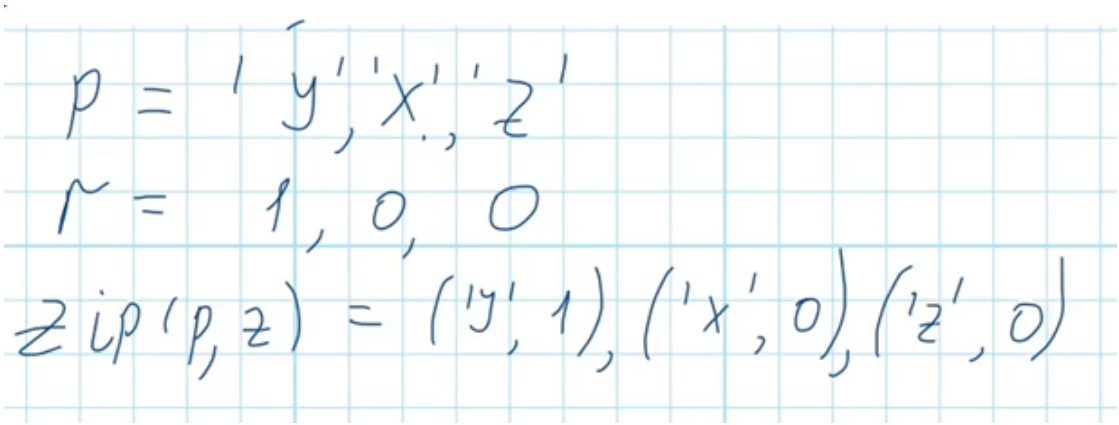
Сама конструкция, заключенная в квадратные скобки, называется генератор списка, или списковое включение. Это инструмент, который позволяет быстро создать какой-либо список.

Внутри этого генератора списка находится синтаксис «r for r in table», в котором перебираются все элементы, записанные ранее в списке table, где r— это строка из данного списка.

Результат перебора этих элементов :

```
[(0, 0, 0), (1, 0, 0), (1, 1, 0)]
[(0, 0, 0), (1, 0, 0), (1, 1, 0)]
[(0, 0, 0), (1, 0, 0), (1, 1, 0)]
[(0, 0, 0), (1, 0, 0), (1, 1, 0)]
[(0, 0, 0), (1, 0, 0), (1, 1, 0)]
[(0, 0, 0), (1, 0, 0), (1, 1, 0)]
```

Для решения задачи необходимо полученные строки связать с переменными x, y, z, в строго определенном порядке. Для этого применяется функция zip. zip(p,r) - попарно объединяет элементы двух списков. Элементы списка букв [x,y,z] и элементы списков значений в строках таблицы.



Для демонстрации работы этой функции применим метод list

```
for p in permutations('xyz'):

    #if [f(**dict(zip(p,r))) for r in table] == [1,1,1]:

        print([list(zip(p,r)) for r in table])
```

и получим следующий результат:

```
[[('x', 0), ('y', 0), ('z', 0)], [('x', 1), ('y', 0), ('z', 0)], [('x', 1), ('y', 1), ('z', 0)]]
[[('x', 0), ('z', 0), ('y', 0)], [('x', 1), ('z', 0), ('y', 0)], [('x', 1), ('z', 1), ('y', 0)]]
[[('y', 0), ('x', 0), ('z', 0)], [('y', 1), ('x', 0), ('z', 0)], [('y', 1), ('x', 1), ('z', 0)]]
[[('y', 0), ('z', 0), ('x', 0)], [('y', 1), ('z', 0), ('x', 0)], [('y', 1), ('z', 1), ('x', 0)]]
[[('z', 0), ('x', 0), ('y', 0)], [('z', 1), ('x', 0), ('y', 0)], [('z', 1), ('x', 1), ('y', 0)]]
[[('z', 0), ('y', 0), ('x', 0)], [('z', 1), ('y', 0), ('x', 0)], [('z', 1), ('y', 1), ('x', 0)]]
```

Каждая из полученных строк отличается порядком переменных. Так в первой строке порядок ‘xyz’, во второй - ‘xzy’, третьей - ‘yxz’ и т.д. Каждая из переменных сцеплена со своим значением, например, (‘x’, 0). Набор из трёх сцепленных пар в квадратных скобках соответствует одной строке таблицы, а целиком каждая из выведенных строк, заключенных во внешние квадратные скобки, варианту таблицы. Как ранее мы выяснили, в результате перестановок по условию этой задачи мы можем получить 6 различных вариантов размещения букв, из которых один будет соответствовать правильному решению.

Т.к. результат работы zip(p,r) невозможно использовать в нашей функции f(x,y,z) непосредственно, из него нужно создать словарь dict(). Словарь - это ключ значения для сопоставления пар.

```
[{'x': 0, 'y': 0, 'z': 0}, {'x': 1, 'y': 0, 'z': 0}, {'x': 1, 'y': 1, 'z': 0}]
[{'x': 0, 'z': 0, 'y': 0}, {'x': 1, 'z': 0, 'y': 0}, {'x': 1, 'z': 1, 'y': 0}]
[{'y': 0, 'x': 0, 'z': 0}, {'y': 1, 'x': 0, 'z': 0}, {'y': 1, 'x': 1, 'z': 0}]
[{'y': 0, 'z': 0, 'x': 0}, {'y': 1, 'z': 0, 'x': 0}, {'y': 1, 'z': 1, 'x': 0}]
[{'z': 0, 'x': 0, 'y': 0}, {'z': 1, 'x': 0, 'y': 0}, {'z': 1, 'x': 1, 'y': 0}]
[{'z': 0, 'y': 0, 'x': 0}, {'z': 1, 'y': 0, 'x': 0}, {'z': 1, 'y': 1, 'x': 0}]
```

В конечном итоге, в каждом из шести вариантов перестановок столбцов мы имеем правильно сопоставленные пары переменных с их числовыми значениями.

Для того, чтобы содержимое словаря dict(), мы могли использовать, как отдельные значения, необходимо его «распаковать». Для этого нужно перед dict() указать два символа *.

```
f(**dict(zip(pr)))
```

Таким образом, в процессе работы этого алгоритма рассматриваются разные варианты перестановок букв переменных, и для каждого варианта этих перестановок вычисляется значение строки.

```
from itertools import *
def f(x,y,z):
    return (not x and y and z) or (not x and not z)
table = [(0,0,0), (1,0,0), (1,1,0)]
for p in permutations('xyz'):
  print([f(**dict(zip(p,r))) for r in table])
  if [f(**dict(zip(p,r))) for r in table] == [1,1,1]:
        print(p)
```

Результат работы программы с выводом значений функции от всех вариантов перестановок переменных будет следующий:

```
[True, False, False]
[True, False, False]

[True, True, False]

[True, True, 1]
('y', 'z', 'x')
[True, False, False]

[True, False, 1]
```

Итоговое решение задачи:

```
from itertools import *
def f(x,y,z):
    return (not x and y and z) or (not x and not z)
table = [(0,0,0), (1,0,0), (1,1,0)]
for p in permutations('xyz'):
```

```
    if [f(**dict(zip(p,r))) for r in table] == [1,1,1]:

        print(p)
```

Результат работы программы: ('y', 'z', 'x')

Ответ: yzx

Задача №2 (3648)

(Е. Джобс) Логическая функция F задаётся выражением (a → d) ∧ ¬(b → c).

?	?	?	?	F
1	0	1	0	1
1	1	1	0	1
0	0	1	0	1

На рисунке приведён частично заполненный фрагмент таблицы истинности функции F, содержащий неповторяющиеся строки. Определите, какому столбцу таблицы истинности функции F соответствует каждая из переменных a, b, c, d.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=31m15s

Решение

Решение этой подобно решению предыдущего задания.

```
# подключение библиотеки функций для работы с итерируемыми объектами
from itertools import *
# возврат значения логической функции
def f(a,b,c,d):
    return (a <= d) and not (b<=c)
# создание таблицы из трех строк через список из кортежей
table = [(1,0,1,0),(1,1,1,0),(0,0,1,0)]
# перебор перестановок


for p in permutations('abcd'):
#проверка условия. Если значения функции от распакованных переменных в #порядке букв,
который задаётся циклом равны значениям функции в заданной #таблице, f=[0,0,1,0]

if [f(**dict(z,ip(p,r))) for r in table] == [1,1,1]:
        print(p)
```

Результат работы программы: ('d', 'a', 'b', 'c')

Ответ: dabc

Задача № 3 (61)

(№ 61) Логическая функция F задаётся выражением (x ∨ y) ∧ (¬x ∨ y ∨ ¬z).

?	?	?	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Определите, какому столбцу таблицы истинности функции F соответствует каждая из переменных x, y, z.

В ответе напишите буквы x, y, z в том порядке, в котором идут соответствующие им столбцы (без разделителей).

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=40m15s

Решение

Для того чтобы сопоставить переменные иногда бывает необязательно рассматривать все строки таблицы. Т.е. для вывода правильного однозначного ответа достаточно меньшего, нежели представлено в исходной таблице количества строк. Если в выбранных строках информации будет недостаточно, в результате работы программы будет получен неоднозначный ответ, т.е. несколько перестановок, соответствующих значениям заданной логической функции.

Например, если для решения этой задачи использовать первые три строки таблицы:

```
from itertools import *


def f(x,y,z):
    return (x or y) and (not x or y or not z)


table = [(0,0,0),(0,0,1),(0,1,0)]


for p in permutations('xyz'):
    if [f(**dict(zip(p,r))) for r in table] == [0,0,1]:
```

```
        print(p)
```

В результате работы программы мы получим два ответа:

```
('x', 'y', 'z')
('y', 'x', 'z')
```

По которым нельзя однозначно определить положение переменных x и y.

При этом, при использовании четырех строк таблицы мы получим единственный, являющийся верным ответ.

Итоговое решение задачи:

```
# подключение библиотеки функций для работы с итерируемыми объектами
from itertools import *
# возврат значения логической функции от трёх переменных
def f(x,y,z,):
    return (x or y) and (not x or y or not z,)
# создание таблицы из первых 4-х строк
table = [(0,0,0),(0,0,1),(0,1,0),(0,1,1)]
# проверка условия. Если значения функции от распакованных переменных в #порядке букв,
который задаётся циклом равны значениям функции в заданной #таблице, f=[0,0,1,0]

for p in permutations('xyz,'):
    if [f(**dict(z,ip(p,r))) for r in table] == [0,0,1,0]:
        print(p)
```

Результат работы программы: ('y', 'x', 'z,')

Ответ: yxz

Задача № 4 (1636)

(№ 1636) Логическая функция F задаётся выражением (x ∨ y) ∧ ¬(y ≡ z) ∧ ¬w.

?	?	?	?	F
1		1		1
0	1		0	1
	1	1	0	1

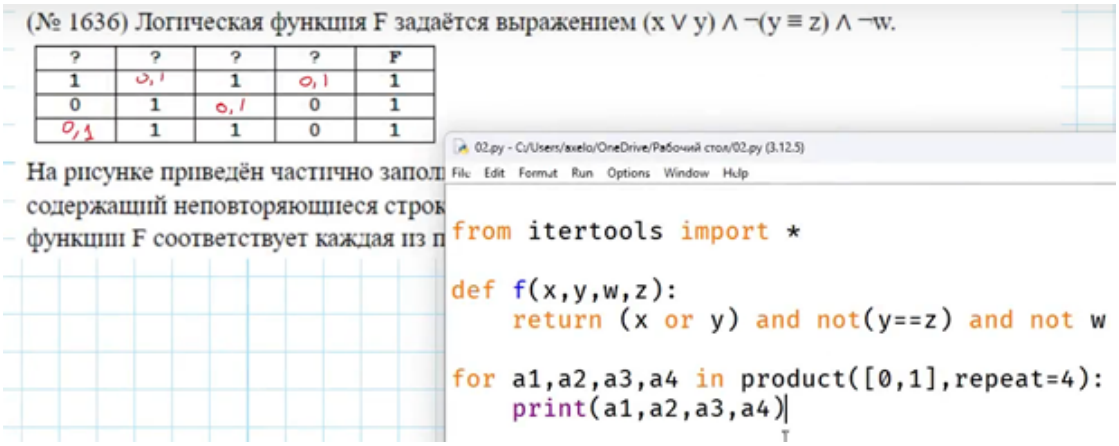
На рисунке приведён частично заполненный фрагмент таблицы истинности функции F, содержащий неповторяющиеся строки. Определите, какому столбцу таблицы истинности функции F соответствует каждая из переменных x, y, z, w.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=46m40s

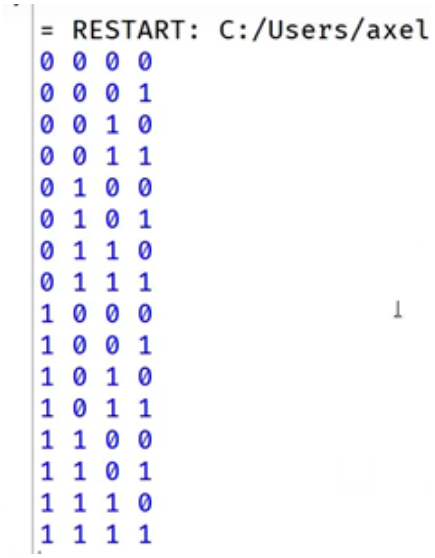
Решение

Обратим внимание на то, что в этой задаче нужно установить порядок переменных в столбцах таблицы, внутри которой имеются пустые места. Это условие можно рассматривать, как дополнительную задачу, для решения которой нужно выяснить, какие значения находятся на месте пропусков.

Для этого мы организуем еще один дополнительный перебор, в котором будем рассмотрим все возможные варианты заполнения этих пустых мест. Т.к. в таблице находится 4 пустоты, будет необходимо перебрать все варианты комбинаций из нулей единиц длины 4. Таких комбинаций будет 16 штук. Все 16 комбинаций мы проверим, для каждого из них посмотрим перестановки и только в одном случае получим ситуацию, когда значения функции будут совпадать с заданными в таблице. Для перебора вариантов заполнения пустых мест будем использовать функцию product(). Обозначим возможные значения пустых мест переменными a1, a2, a3, a4. Т.к. эти значения могут быть нулями или единицами зададим соответствующие параметры перебора product ([0, 1], repeat =4). Значения 0 и 1 заключены в квадратные скобки т.к. представляют собой элементы списка.



В результате мы получим 16 вариантов комбинаций из нулей и единиц для пустых мест в таблице.



Далее решение задачи будет осуществляться похожим на предыдущие примеры образом, с тем отличием, что в при создании таблицы в скобках соответствующих строкам для обозначения пустых мест мы будем использовать переменные a1, a2, a3, a4.

```
from itertools import *
```

```
def f(x,y,w,z):
    return (x or y) and not(y==z) and not w

for a1,a2,a3,a4 in product([0,1],repeat=4):
    table = [(1,a1,1,a2),(0,1,a3,0),(a4,1,1,0)]
    print(table)
```

```
[(1, 0, 1, 0), (0, 1, 0, 0), (0, 1, 1, 0)]
[(1, 0, 1, 0), (0, 1, 0, 0), (1, 1, 1, 0)]
[(1, 0, 1, 0), (0, 1, 1, 0), (0, 1, 1, 0)]
[(1, 0, 1, 0), (0, 1, 1, 0), (1, 1, 1, 0)]
[(1, 0, 1, 1), (0, 1, 0, 0), (0, 1, 1, 0)]
[(1, 0, 1, 1), (0, 1, 0, 0), (1, 1, 1, 0)]
[(1, 0, 1, 1), (0, 1, 1, 0), (0, 1, 1, 0)]
[(1, 0, 1, 1), (0, 1, 1, 0), (1, 1, 1, 0)]
[(1, 1, 1, 0), (0, 1, 0, 0), (0, 1, 1, 0)]
[(1, 1, 1, 0), (0, 1, 0, 0), (1, 1, 1, 0)]
[(1, 1, 1, 0), (0, 1, 1, 0), (0, 1, 1, 0)]
[(1, 1, 1, 0), (0, 1, 1, 0), (1, 1, 1, 0)]
[(1, 1, 1, 1), (0, 1, 0, 0), (0, 1, 1, 0)]
[(1, 1, 1, 1), (0, 1, 0, 0), (1, 1, 1, 0)]
[(1, 1, 1, 1), (0, 1, 1, 0), (0, 1, 1, 0)]
[(1, 1, 1, 1), (0, 1, 1, 0), (1, 1, 1, 0)]
```

В задаче указано, что все строки таблицы должны быть различными, поэтому при решении, нам необходимо предусмотреть способ исключения таблиц с повторяющимися строками. Следовательно, нужно предусмотреть условие, которое будет отслеживать совпадающие строки. Для этого мы будем сравнивать длину списка «таблица» с длиной множества из элементов списка «таблица». Т.к. множество сохраняет только уникальные элементы, в случае, если среди строк проверяемого варианта таблицы окажутся одинаковые, длина множества списка «таблицы» будет меньше длины самого списка. Условие будет выглядеть следующим образом:

```
if len(table) == len(set(table)):
```

Если это условие выполняется, т.е. в списке столько же элементов, сколько уникальных элементов в его множестве, значит все строки в таблице различны.

Таких вариантов в этой задаче меньше первоначальных шестнадцати (10 шт.):

```
('z', 'y', 'x')
('z', 'y', 'x')
('z', 'y', 'x')
('z', 'y', 'x')
('z', 'y', 'x')
('z', 'y', 'x')
```

```
('z', 'y', 'x')
('z', 'y', 'x')
('z', 'y', 'x')
('z', 'y', 'x')
```

После того, как проверено это условие, дальнейшее решение будет аналогичным решениям предыдущих задач.

Итоговое решение задачи:

```
# подключение библиотеки функций для работы с итерируемыми объектами
from itertools import *
# возврат значения логической функции
def f(x,y,w,z,):
    return (x or y) and not(y==z,) and not w
# заполнение четырёх пропусков таблицы, для этого перебираем все #комбинации 0 и 1
длины равной количеству пропусков
for a1, a2, a3, a4 in product([0,1],repeat=4):
#на основании комбинаций значений a1…a4 строим таблицу из трёх строк
    table = [(1,a1,1,a2),(0,1,a3,0),(a4,1,1,0)]
#проверяем, что полученные строки таблицы различны, т.е. длина таблицы #равна длине
списка
    if len(table) == len(set(table)):
#проверка значений строк
        for p in permutations('xywz,'):
            if [f(**dict(z,ip(p,r))) for r in table]==[1,1,1]:
                print(p)
```

Результат работы программы: ('z,', 'y', 'x', 'w')

Ответ: zyxw

Задача №5 (6210)

(А. Богданов) Логическая функция F задаётся выражением ¬(((¬w → ¬y) → ¬z) → x). На рисунке приведён частично заполненный фрагмент таблицы истинности функции F, содержащий неповторяющиеся строки. Определите, какому столбцу таблицы истинности функции F соответствует каждая из переменных x, y, z, w.

?	?	?	?	F
		1	0	1
	1		1	1
0	1		0	0

В ответе напишите буквы x, y, z, w в том порядке, в котором идут соответствующие им столбцы. Буквы в ответе пишите подряд, никаких разделителей между буквами ставить не нужно.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=1h1m15s

Решение

Как и в предыдущем задании в этой задаче имеется дополнительное условие в виде пустот в таблице. В данном случае в таблице пять пустых мест. В остальном решение задачи будет аналогичным предыдущей, но следует обратить внимание на расстановку дополнительных скобок в логическом выражении для соблюдения выполнения приоритета операций.

```
# подключение библиотеки функций для работы с итерируемыми объектами
from itertools import *
# возврат значения логической функции

def f(x,y,w,z):
#для корректного выполнения выражения с учетом приоритета логических операций
#следует расставить скобки, согласно логическому приоритету

    return not((((not w) <= (not y))<= (not z)) <= x)
# заполнение пяти пропусков таблицы, для этого перебираем все #комбинации 0 и 1 длины равной количеству пропусков
for a1, a2,a3,a4,a5 in product([0,1],repeat=5):
#на основании комбинаций значений a1…a5 строим таблицу из трёх строк
    table = [(a1, a2,1,0),(a3,1,a4,1),(0,1,a5,0)]
#проверяем, что полученные строки таблицы различны, т.е. длина таблицы #равна длине списка
    if len(table)==len(set(table)):
        for p in permutations('xywz'):
#проверка значений строк
            if [f(**dict(zip(p,r))) for r in table]==[1,1,0]:
                print(p)
```

Результат работы программы: ('x', 'z,', 'w', 'y')

('x', 'z,', 'w', 'y')

Т.к. оба ответа совпадают, задача решена верно. Подобный вывод результата программы говорит о том, что в одной из строк таблицы от перемены значений в пустотах значение функции не поменяется.

Ответ: xzwy

Задание №6 (6366)

Логические функции F1 и F2 задаются выражениями
$F1 = (w \to y) \equiv (z \to x)$, $F2 = (w \to y) \wedge (\neg x \equiv z)$.
На рисунке приведён частично заполненный фрагмент таблицы истинности этих функций, содержащий неповторяющиеся строки. Определите, какому столбцу таблицы истинности соответствует каждая из переменных x, y, z, w.

?	?	?	?	F1	F2
0		0	0	0	1
0	0	0		0	
0	1	1			0

В ответе напишите буквы x, y, z, w в том порядке, в котором идут соответствующие им столбцы. Буквы в ответе пишите подряд, никаких разделителей между буквами ставить не нужно

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=1h12m15s

Решение

В этой задаче один и тот же набор значений переменных подставляется в одну функцию и во вторую функцию, следовательно, в решении мы будем выполнять все действия для двух функций.

```
from itertools import *

def f1(x,y,w,z):

    return (w<=y) == (z<= x)

def f2(x,y,w,z):
#для корректного выполнения выражения с учетом приоритета операции #отрицания следует добавить скобки, перед операцией эквиваленции


    return (w<=y) and ((not x) == z)
# заполнение пяти пропусков таблицы, для этого перебираем все
#комбинации 0 и 1 длины равной количеству пропусков
for a1, a2,a3,a4,a5 in product([0,1],repeat=5):
    table = [(0, a1,0,0),(0,0,0,a2),(0,1,1,a3)]
#проверяем, что полученные строки таблицы различны,
    if len(table)==len(set(table)):
#отдельно проверяем значения строк для каждой функции

        for p in permutations('xywz'):
            if [f1(**dict(zip(p,r))) for r in table] == [0,0,a4] and \
               [f2(**dict(zip(p,r))) for r in table] == [1,a5,0]:
```

```
                print(p)
```

Результат работы программы: ('y', 'z', 'x', 'w')

('y', 'z', 'x', 'w')

Полученные два ответа следствие того, что таблицу можно заполнить двумя разными способами, и оба способа дают правильный ответ.

Ответ: yzxw

Ответ:

Задание № 7(6616)

(Е. Джобс) Логические функции F1 и F2 задаются выражениями

$F1 = (x \to y) \vee (\neg w \equiv z)$, $F2 = (x \to y) \equiv (w \wedge \neg z)$.

На рисунке приведёны частично заполненные три различные строки таблицы истинности этих функций, в которых значения функций F1 и F2 равны.

?	?	?	?	F_1	F_2
			0	p	p
		0	0	q	q
	0	0	0	r	r

В ответе напишите буквы x, y, z, w в том порядке, в котором идут соответствующие им столбцы. Буквы в ответе пишите подряд, никаких разделителей между буквами ставить не нужно.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=1h18m50s

Решение

В таблице этой задачи в столбцах F1 и F2 указаны одинаковые буквы значения, обозначающие равенство значений заданных логических функций. Для решения в условии проверки строк сравним значения f1 и f2. Если они совпадают выведем результат.

```
from itertools import *

def f1(x,y,w,z):
    return (x<=y) or ((not w) == z)

def f2(x,y,w,z):
    return (x<=y) == (w and (not z))
```

```
for a1,a2,a3,a4,a5,a6 in product([0,1],repeat=6):
    table = [(a1,a2,a3,0),(a4,a5,0,0),(a6,0,0,0)]
#проверяем, что полученные строки таблицы различны,
    if len(table)==len(set(table)):
        for p in permutations('xywz'):
            if [f1(**dict(zip(p,r))) for r in table] == \
               [f2(**dict(zip(p,r))) for r in table]:
                print(p)
```

Результат работы программы: ('w', 'y', 'x', 'z')

Ответ: wyxz

```
#для аналитического решения
from itertools import *
#строим таблицу
for x,y,w,z in product([0,1],repeat=4):
    f1 = (x<=y) or ((not w ==z)
    f2 = (x<=y) == (w and not z)
# Если значения функций совпадают выводим строки таблицы
    if f1==f2:
        print(x,y,w,z,f1,f2)
```

Результат работы программы: 0 0 1 0 True True

0 1 1 0 True True

1 0 0 1 True True

1 1 1 0 True True

Сравним исходную и полученную таблицы

Ответ: wyxz

Задание №8 (3974)

(В.Н. Шубинкин) Логическая функция F задаётся выражением x ∧ (y → z) ∨ w. Ниже приведён частично заполненный фрагмент таблицы истинности этой функции, содержащий неповторяющиеся строки. Сколькими способами можно поставить в соответствие переменные w, x, y, z столбцам таблицы истинности функции F, опираясь на информацию из данного фрагмента?

?	?	?	?	F
1	0		1	0
	0	1		0
	0			0

Пример. Функция F задана выражением x ∨ y ∨ z, а фрагмент таблицы истинности имеет вид:

?	?	?	F
0	1	1	1

В этом случае переменные можно расставить любым способом, значит, ответом будет число 6.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241189?t=1h27m40s

Решение

Отличие этой задачи от предыдущих в том, что в ней по условию известно, что распределение букв в таблице не является единственно возможным, и необходимо выяснить сколько таких вариантов существует.

Первый вариант решения задачи, частично аналитический. Проанализируем полученную с помощью кода таблицу. Уберем повторяющиеся строки. Получим ответ.

(№ 3974) (В.Н. Шубинкин) Логическая функция F задаётся выражением x ∧ (y → z) ∨ w. Ниже приведён частично заполненный фрагмент таблицы истинности этой функции, содержащий неповторяющиеся строки. Сколькими способами можно поставить в соответствие переменные w, x, y, z столбцам таблицы истинности функции F, опираясь на информацию из данного фрагмента?

?	?	?	?	F
1	0		1	0
	0	1		0
	0			0

```
('x', 'w', 'z', 'y')
('y', 'w', 'z', 'x')
('x', 'w', 'z', 'y')
('x', 'w', 'z', 'y')
('y', 'w', 'z', 'x')
('y', 'w', 'z', 'x')
('x', 'w', 'z', 'y')
('z', 'w', 'x', 'y')
('x', 'w', 'z', 'y')
('z', 'w', 'x', 'y')
('x', 'w', 'z', 'y')
('z', 'w', 'x', 'y')
('y', 'w', 'x', 'z')
('y', 'w', 'z', 'x')
('y', 'w', 'x', 'z')
('y', 'w', 'z', 'x')
('y', 'w', 'x', 'z')
('y', 'w', 'z', 'x')
```

```
from itertools import *

def f(x,y,w,z):
    return x and (y<=z) or w

for a1,a2,a3,a4,a5,a6 in product([0,1],repeat=6):
    table=[(1,0,a1,1),(a2,0,1,a3),(a4,0,a5,a6)]
    if len(table)==len(set(table)):
        for p in permutations('xywz'):
            if [f(**dict(zip(p,r))) for r in table]==[0,0,0]:
                print(p)
```

4

Ответ: 4

Чтобы полностью автоматизировать решение можно доработать наш код, добавив в программу переменную «список», и, использовав функцию множества, set(). Добавить элементы в множества можно с помощью команды add(). Чтобы сразу получить ответ используем len()

Полностью автоматизированное решение задачи:

```
from itertools import *

def f(x,y,w,z,):
    return x and (y<=z,) or w
#задаем переменную для множества уникальных значений
spisok = set()
for a1,a2,a3,a4,a5,a6 in product([0,1],repeat=6):
#строим таблицу
    table=[(1,0,a1,1),(a2,0,1,a3),(a4,0,a5,a6)]
#проверяем, что полученные строки таблицы различны,
    if len(table)==len(set(table)):
#проверка значений строк
        for p in permutations('xywz'):
            if [f(**dict(z,ip(p,r))) for r in table]==[0,0,0]:
#добавляем уникальные варианты ответов в список, являющийся множеством
                spisok.add(p)
#выводим длину полученного множества
print(len(spisok))
```

Результат работы программы: 4

Ответ: 4