. Текстовый разбор домашки 3

DZ_232_prog_1

Задача № 1 (886)

Исполнитель Калькулятор преобразует число на экране. У исполнителя есть три команды, которым присвоены номера:

1. Прибавить 1
2. Прибавить 4
3. Умножить на 2

Сколько существует программ, состоящих из 7 команд, для которых при исходном числе 3 результатом является число 27?

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h0m0s

Решение

Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- step: количество выполненных команд.

Рекурсивно она будет вычислять возможные пути от c до e.

Условия задачи:

1. Если текущее число c больше целевого числа e, или количество команд step больше 7, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, выполнилось ли ровно 7 шагов. Если шагов было ровно 7, возвращается 1 (это значит, что найдено одно решение). Если шагов больше или меньше, возвращается 0, потому что это не удовлетворяет условиям задачи (нужно именно 7 шагов).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:
   - Увеличить c на 1,
   - Увеличить c на 4,
   - Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и увеличенного шага (то есть step+1). Результаты всех этих вызовов суммируются. Выводим на экран результат работы функции при c=3, e=27 и step = 0. То есть считаем количество траекторий из 3 в 27 при стартовом количестве команд 0.

```
def f(c,e,step):
    if c>e or step>7: return 0
    if c==e: return step==7
    if c<e: return f(c+1,e,step+1)+f(c+4,e,step+1)+f(c*2,e,step+1)

print(f(3,27,0))
```

Ответ: 37

Telegram: @fast_ege

DZ_232_prog_2

Задача №2 (586)

Исполнитель Простачок преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:
1. Прибавить 2
2. Прибавить 3
3. Умножить на 2
Первая команда увеличивает число на 2, вторая – на 3, третья – увеличивает число вдвое.

Сколько чисел может быть результатом работы алгоритма для входного значения 10, если известно, что в алгоритме 5 команд?Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h2m20s

Решение
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:
- c: текущее число.
- e: конечное число, к которому нужно прийти.
- s: число выполненных команд.

Рекурсивно она будет вычислять возможные пути от c до e.
Условия задачи:
1. Если текущее число c больше целевого числа e, или количество команд s больше 5, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, выполнилось ли ровно 5 шагов. Если шагов было ровно 5, возвращается 1 (это значит, что найдено одно решение). Если шагов больше или меньше, возвращается 0, потому что это не удовлетворяет условиям задачи (нужно именно 5 шагов).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:
- Увеличить c на 2,
- Увеличить c на 3,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и увеличенного шага (то есть s+1). Результаты всех этих вызовов суммируются.
Сделаем счетчик k.
Будем перебирать числа от 11 до 400 (максимально возможный результат, если число 10 пять умножить на 2, равен 320, поэтому 400 нам подходит). Если f (10,e,0) >0, то есть существует хоть одна траектория от 10 до e, то увеличиваем счетчик k на 1.
Выводим k, это ответ на задачу.

```
def f(c,e,s):
    if c>e or s>5: return 0
    if c==e: return s==5
    if c<e: return f(c+2,e,s+1)+f(c+3,e,s+1)+f(c*2,e,s+1)

k = 0
for e in range(11,400):
    if f(10,e,0)>0:
        k += 1
print(k)
```

Ответ: 83
Telegram: @fast_ege

DZ_232_prog_3

Задача № 3 (4112)

Исполнитель преобразует число, записанное на экране. У исполнителя есть две команды, которым присвоены номера:

1. Прибавить 2
2. Умножить на 2

Первая команда увеличивает число на экране на 2, вторая – вдвое.

Программа для исполнителя – это последовательность команд. За какое минимальное количество команд Исполнитель может получить из числа 1 число 100.

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h4m35s

Решение

Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- s: число выполненных команд.

Рекурсивно она будет вычислять возможные пути от c до e.

Условия задачи:

1. Если текущее число c больше целевого числа e, или количество команд s больше некого количества mx, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, выполнилось ли ровно mx шагов. Если шагов было ровно mx, возвращается 1 (это значит, что найдено одно решение). Если шагов больше или меньше, возвращается 0, потому что это не удовлетворяет условиям задачи (нужно именно mx шагов).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для двух действий:

- Увеличить c на 2,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и увеличенного шага (то есть s+1). Результаты всех этих вызовов суммируются.

Будем перебирать числа от 1 до 20. Если f (1,100,0) >0, то есть существует хоть одна траектория от 1 до 100, то выводим mx. Получаем список значений, минимальное из который равно 7. это ответ на задачу.

```
def f(c,e,s):
    if c>e or s>mx: return 0
    if c==e: return s==mx
    if c<e: return f(c+2,e,s+1)+f(c*2,e,s+1)

for mx in range(1,20):
    if f(1,100,0)>0:
        print(mx)
```

Ответ: 7

Telegram: @fast_ege

DZ_232_prog_4

Задача № 4 (3032)

Исполнитель Калькулятор преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:

1. Прибавь 1

2. Прибавь 3
3. Умножь на 2
Первая команда увеличивает число на экране на 1, вторая увеличивает его на 3, третья – умножает на 2. Программа для исполнителя – это последовательность команд. Сколько существует программ, которые преобразуют исходное число 3 в число 30 и при этом не содержат двух команд «Прибавить 1» подряд?
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h6m30s
Решение
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- p: некая команда из имеющихся у исполнителя

Рекурсивно она будет вычислять возможные пути от c до e.
Условия задачи:
1. Если текущее число c больше целевого числа e, то вернуть 0.
2. Если текущее число c равно целевому числу e, то вернуть 1.
3. Если текущее число c меньше целевого числа e, и команда p – это команда «прибавить 1» (‘+1’), то вычисляются возможные пути для двух действий:

- Увеличить c на 3,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и команд «прибавить 3» (‘+3’) и «умножить на 2» (‘*2’) соответственно. Результаты всех этих вызовов суммируются.
4. Если текущее число c меньше целевого числа e, и команда p не является командой «прибавить 1» (‘+1’), то вычисляются возможные пути для трех действий:

- Увеличить c на 1,
- Увеличить c на 3,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и команд «прибавить 1» (‘+1’), «прибавить 3» (‘+3’) и «умножить на 2» (‘*2’) соответственно. Результаты всех этих вызовов суммируются.
Выводим на экран результат работы функции при c=3, e=30 и p = ‘ ’. То есть считаем количество траекторий из 3 в 30, при том что первая команда p может быть любой.

```
def f(c,e,p):
    if c>e: return 0
    if c==e: return 1
    if c<e and p=='+1': return f(c+3,e,'+3')+f(c*2,e,'*2')
    if c<e and p!='+1': return f(c+1,e,'+1')+f(c+3,e,'+3')+f(c*2,e,'*2')

print(f(3,30,''))
```

Ответ: 407
Telegram: @fast_ege

DZ_232_prog_5

Задача №5 (3162)
Исполнитель Калькулятор преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:
1. Прибавь 1
2. Прибавь 2

3. Умножь на 2
Первая команда увеличивает число на экране на 1, вторая увеличивает его на 2, третья – умножает на 2. Программа для исполнителя – это последовательность команд. Сколько существует программ, которые преобразуют исходное число 2 в число 12 и при этом содержат ровно одну команду «Умножь на 2»?
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h9m05s
Решение
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- k: количество применений команды «умножить на 2».

Рекурсивно она будет вычислять возможные пути от c до e.
Условия задачи:
1. Если текущее число c больше целевого числа e, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, выполнилось ли умножение ровно 1 раз. Если умножение выполнилось 1 раз, возвращается 1 (это значит, что найдено одно решение). Если шагов с умножением больше или меньше, возвращается 0, потому что это не удовлетворяет условиям задачи (нужен ровно 1 шаг с умножением).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:

- Увеличить c на 1,
- Увеличить c на 2,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и шага k, если действие не было связано с умножением, и увеличенного шага (то есть k+1), если была выполнена команда «умножить на 2». Результаты всех этих вызовов суммируются.
Выводим на экран результат работы функции при c=2, e=12 и k = 0. То есть считаем количество траекторий из 2 в 12, при том что число k изначально равно 0. Это ответ на задачу.

```
def f(c,e,k):
    if c>e: return 0
    if c==e: return k==1
    if c<e: return f(c+1,e,k)+f(c+2,e,k)+f(c*2,e,k+1)

print(f(2,12,0))
```

Ответ: 68
Telegram: @fast_ege

DZ_232_prog_6

Задание №6 (5443)
Исполнитель преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:
1. Прибавить 1
2. Прибавить 2
3. Умножить на 2
Первая команда увеличивает число на 1, вторая – на 2, третья – вдвое.
Программа для исполнителя – это последовательность команд.
Сколько существует таких программ, которые исходное число 3 преобразуют в число 25 и при этом в программе есть все три команды?

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h11m00s

Решение

Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать пять параметров:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- k1: количество применений команды 1 (прибавить 1).
- k2: количество применений команды 2 (прибавить 2).
- k3: количество применений команды 3 (умножить на 2).

Рекурсивно она будет вычислять возможные пути от c до e.

Условия задачи:

1. Если текущее число c больше целевого числа e, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, содержит ли траектория все три команды (k1>0, k2>0, k3>0). Если это верно, возвращается 1 (это значит, что найдено одно решение). Если это не так, то есть какого из шагов в траектории нет, возвращается 0, потому что это не удовлетворяет условиям задачи (необходимо, что бы в траектории содержали все три шага).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:

- Увеличить c на 1,
- Увеличить c на 2,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и шагов k1, k2, k3 , при этом, если действие связано с шагом k1, то шаг k1 увеличивается на 1, а шаги k2 и k3 остаются без изменения; если действие связано с шагом k2, то шаг k2 увеличивается на 1, а шаги k1 и k3 остаются без изменения если действие связано с шагом k3, то шаг k3 увеличивается на 1, а шаги k1 и k2 остаются без изменения Результаты всех этих вызовов суммируются.

Выводим на экран результат работы функции при c=3, e=25 и k1=0, k2=0, k3=0. То есть считаем количество траекторий из 3 в 25, при том что все числа k изначально равно 0. Это ответ на задачу.

```
def f(c,e,k1,k2,k3):
    if c>e: return 0
    if c==e: return k1>0 and k2>0 and k3>0
    if c<e: return f(c+1,e,k1+1,k2,k3)+f(c+2,e,k1,k2+1,k3)+\
       f(c*2,e,k1,k2,k3+1)

print(f(3,25,0,0,0))
```

Ответ: 15092

Telegram: @fast_ege

DZ_232_prog_7

Задание № 7(5236)

Исполнитель преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:

1. Прибавь 2
2. Умножь на 3
3. Умножь на 4

Выполняя первую из них, исполнитель увеличивает число на экране на 3, выполняя вторую – умножает на 3, выполняя третью – умножает на 4. Программой для исполнителя называется последовательность команд. Сколько существует различных программ, которые преобразуют

исходное число 2 в число 400, и при этом траектория вычислений содержит более 50 различных чисел (без учёта начального и конечного)?
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h13m05s
Решение
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- k: количество чисел в траектории.

Рекурсивно она будет вычислять возможные пути от c до e.
k – это счетчик, увеличиваем его на единицу, если мы попали в какое-то число.
Условия задачи:
1. Если текущее число c больше целевого числа e, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, достигло ли число k значения 52 (содержит ли траектория более 52 различных чисел, включая начальное и конечное). Если k>52, возвращается 1 (это значит, что найдено одно решение). Если k<52, возвращается 0, потому что это не удовлетворяет условиям задачи (нужен что бы траектория вычислений содержала более 52 различных чисел).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:

- Увеличить c на 1,
- Умножить c на 3,
- Умножить c на 4.

Рекурсивно вызывается функция для каждого из этих новых значений числа и значения k.
Результаты всех этих вызовов суммируются.
Выводим на экран результат работы функции при c=2, e=400 и k = 0. То есть считаем количество траекторий из 2 в 400, при том что количество чисел в траектории изначально равно 0. Это ответ на задачу.

```
def f(c,e,k):
    k += 1
    if c>e: return 0
    if c==e: return k>52
    if c<e: return f(c+2,e,k)+f(c*3,e,k)+f(c*4,e,k)

print(f(2,400,0))
```

Ответ: 6142
Telegram: @fast_ege

DZ_232_prog_8

Задание №8 (4492)
Исполнитель преобразует число, записанное на экране.
У исполнителя есть три команды, которым присвоены номера:
Прибавить 1
Прибавить 2
Умножь на 2
Первая из них увеличивает число на экране на 1, вторая увеличивает его на 2, третья увеличивает его в 2 раза.
Программа для исполнителя – это последовательность команд.

Сколько существует таких программ, которые преобразуют исходное число 2 в число 40 и при этом траектория вычислений программы содержит ровно одно нечётное число?
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h15m25s
Решение
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- k1: количество нечетных чисел в траектории.

Рекурсивно она будет вычислять возможные пути от c до e.
Условия задачи:
1. Если число с нечетное, то есть c%2!=0, то увеличиваем k1 на 1.
2. Если текущее число c больше целевого числа e, или количество нечетных чисел k1 то вернуть 0.
3. Если текущее число c равно целевому числу e, то проверяется, равно ли k1 единице (содержит ли траектория ровно одно нечетное число), в этом случае возвращается 1 (это значит, что найдено одно решение). Если число нечетных чисел в траектории отличается от 1, возвращается 0, потому что это не удовлетворяет условиям задачи.
4. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:

- Увеличить c на 1,
- Увеличить c на 2,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и k1. Результаты всех этих вызовов суммируются.
Выводим на экран результат работы функции при c=2, e=40 и k1 = 0. То есть считаем количество траекторий из 2 в 40, при том что число k1 изначально равно 0. Это ответ на задачу.

```
def f(c,e,k1):
    if c%2!=0: k1+=1
    if c>e or k1>1: return 0
    if c==e: return k1==1
    if c<e: return f(c+1,e,k1)+f(c+2,e,k1)+f(c*2,e,k1)

print(f(2,40,0))
```

Ответ: 626
Telegram: @fast_ege

DZ_232_prog_9

Задание №9(2714)
Исполнитель K22 преобразует число, записанное на экране.
У исполнителя есть три команды, которым присвоены номера:
Прибавить 1
Прибавить 3
Прибавить 5
Первая из них увеличивает число на экране на 1, вторая увеличивает его на 3, третья увеличивает его на 5.
Программа для исполнителя K22 – это последовательность команд.
Сколько существует таких программ, которые преобразуют исходное число 3 в число 25 и при этом траектория вычислений программы содержит ровно 6 четных чисел?
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h18m00s

Решение

Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- k0: количество четных чисел в траектории.

Рекурсивно она будет вычислять возможные пути от c до e.

Условия задачи:

1. Если число c четное, то есть c%2==0, то увеличиваем k0 на 1.
2. Если текущее число c больше целевого числа e, то вернуть 0.
3. Если текущее число c равно целевому числу e, то проверяется, равно ли k0 шести (содержит ли траектория ровно шесть четных чисел), в этом случае возвращается 1 (это значит, что найдено одно решение). Если число четных чисел в траектории отличается от 6, возвращается 0, потому что это не удовлетворяет условиям задачи.
4. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:

- Увеличить c на 1,
- Увеличить c на 3,
- Увеличить c на 5.

Рекурсивно вызывается функция для каждого из этих новых значений числа и k0. Результаты всех этих вызовов суммируются.

Выводим на экран результат работы функции при c=3, e=25 и k0 = 0. То есть считаем количество траекторий из 3 в 25, при том что число k0 изначально равно 0. Это ответ на задачу.

```
def f(c,e,k0):
    if c%2==0: k0+=1
    if c>e: return 0
    if c==e: return k0==6
    if c<e: return f(c+1,e,k0)+f(c+3,e,k0)+f(c+5,e,k0)

print(f(3,25,0))
```

Ответ: 3432

Telegram: @fast_ege

DZ_232_prog_10

Задание №10 (4496)

Исполнитель преобразует число на экране.

У исполнителя есть две команды, которым присвоены номера:

1. Прибавить 1
2. Удвоить

Первая команда увеличивает число на экране на 1, вторая – умножает его на 2.

Программа для исполнителя – это последовательность команд.

Например, программа 121 при исходном числе 3 последовательно получит числа 4, 8 и 9.

Результатом программы будет число 9.

Какое минимальное натуральное число нельзя получить из исходного числа 1 после выполнения программы, содержащей не более 5 команд?

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h19m45s

Решение

Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- s: число выполненных команд.

Рекурсивно она будет вычислять возможные пути от c до e.
Условия задачи:
1. Если текущее число c больше целевого числа e, или количество команд s больше 5, то вернуть 0.
2. Если текущее число c равно целевому числу e, то проверяется, выполнилось ли ровно 5 шагов. Если шагов было 5 или меньше, возвращается 1 (это значит, что найдено одно решение). Если шагов больше, возвращается 0, потому что это не удовлетворяет условиям задачи (нужно не более 5 шагов).
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для двух действий:

- Увеличить c на 1,
- Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа и увеличенного шага (то есть s+1). Результаты всех этих вызовов суммируются.
Будем перебирать числа x от 2 до 100. Если f (1, x, 0) == 0, то есть не существует ни одной траектории от начального числа 1 до числа x, содержащей меньше 5 шагов, то выводим x на экран. Получаем список чисел, наименьшее число в этом списке это ответ на задачу.

```
def f(c,e,s):
    if c>e or s>5: return 0
    if c==e: return s<=5
    if c<e: return f(c+1,e,s+1)+f(c*2,e,s+1)

for x in range(2,100):
    if f(1,x,0)==0:
        print(x)
```

Ответ: 15
Telegram: @fast_ege

DZ_232_prog_11

Задача № 11(7011)
Исполнитель Калькулятор преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены коды:
A. Прибавь 2
B. Прибавь 3
C. Умножь на 2
Первая команда увеличивает число на 2 раза, вторая – на 3 раза, третья – в 2 раза. Программа для исполнителя – это последовательность команд. Сколько существует программ, для которых при исходном числе 2 результатом будет являться число 40, при этом траектория вычисления не содержит число 28, а также не содержит подпоследовательность команд BACA. Траектория вычисления программы – это последовательности результатов выполнения всех команд.
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h21m50s
Решение
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- tr: некая последовательность команд из имеющихся у исполнителя.

Рекурсивно она будет вычислять возможные пути от c до e.
Условия задачи:
1. Если текущее число c больше целевого числа e, или c равно 28, или последовательность команд tr содержит последовательность BACA, то вернуть 0.
2. Если текущее число c равно целевому числу e, то вернуть 1.
3. Если текущее число c меньше целевого числа e, то вычисляются возможные пути для трех действий:
· Увеличить c на 2,
· Увеличить c на 3,
· Умножить c на 2.
Рекурсивно вызывается функция для каждого из этих новых значений числа и последовательности команд соответственно: в первом случае в последовательность tr добавляется команда “A”, во втором случае – команда “B”, и в третьем – команда “C”. Результаты всех этих вызовов суммируются.
Выводим на экран результат работы функции при c=2, e=40 и tr = ‘ ’. То есть считаем количество траекторий из 2 в 40, при том что изначально последовательность tr не содержит ни одной команды.

```
def f(c,e,tr):
    if c>e or c==28 or 'BACA' in tr: return 0
    if c==e: return 1
    if c<e: return f(c+2,e,tr+'A')+f(c+3,e,tr+'B')+f(c*2,e,tr+'C')

print(f(2,40,''))
```

Ответ: 27609
Telegram: @fast_ege

DZ_232_prog_12

Задача № 12 (5926)
Исполнитель Калькулятор преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:
А. Прибавить 1.
Б. Прибавить 7.
С. Умножить на 4.
Первая команда увеличивает число на 1, вторая – на 7, третья – умножает на 4. Сколько различных результатов можно получить из исходного числа 1 после выполнения программы, содержащей 24 команд, если известно, что запрещено повторять команду, сделанную на предыдущем шаге.
Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h19m25s
Решение
Создадим пустое множество d, в которое будем добавлять числа, получающиеся после выполнения 24 команд.
Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать три параметра:
- c: текущее число.
- s: количество команд.
- p: предыдущие выполненные команды.

Рекурсивно она будет вычислять возможные пути от c, до чисел, которые можно достичь за 24 шага. Если количество шагов достигло 24, добавляем получившееся число в множество d и возвращаем 1, это значит, что найдено одно решение.

Если число шагов не достигло 24, то рассматриваем возможные случаи:

· На предыдущем шаге выполнялась команда “+1”, тогда суммируем результаты работы рекурсивных функций: f, увеличивающей текущее число с на 7, количество команд на 1, и выполненная команда p меняется на “+7” и f, увеличивающей текущее число с в 4 раза, количество команд на 1, при этом выполненная команда p меняется на “*4”.

· На предыдущем шаге выполнялась команда “+7”, тогда суммируем результаты работы рекурсивных функций: f, увеличивающей текущее число с на 1, количество команд на 1, и выполненная команда p меняется на “+1” и f, увеличивающей текущее число с в 4 раза, количество команд на 1, при этом выполненная команда p меняется на “*4”.

· На предыдущем шаге выполнялась команда “*4”, тогда суммируем результаты работы рекурсивных функций: f, увеличивающей текущее число с на 1, количество команд на 1, и выполненная команда p меняется на “+1” и f, увеличивающей текущее число с на 7, количество команд на 1, и выполненная команда p меняется на “+7”.

· Команда запускается впервые, то есть на предыдущем шаге команд не было: p=‘ ’, тогда суммируем результаты работы всех трех рекурсивных функций: f, увеличивающей текущее число с на 1, количество команд на 1, и выполненная команда p меняется на “+1”, f, увеличивающей текущее число с на 7, количество команд на 1, и выполненная команда p меняется на “+7” и f, увеличивающей текущее число с в 4 раза, количество команд на 1, при этом выполненная команда p меняется на “*4”.

Выводим на экран результат работы функции при c=1, s=0 и p= ' '. То есть количество траекторий из 1, которые можно пройти за 24 шага, при этом начальное количество шагов у нас равно 0 и предыдущая команда пустая, так как ранее функция не запускалась. Так же выведем на экран длину множества d, которая будет являться ответом на задачу. Нужно обратить внимание, что количество траекторий очень отличается от длины множества d, потому что в множество d добавляются только различные числа.

```
d = set()
def f(c,s,p):
    if s==24:
        d.add(c)
        return 1
    if s<24:
        if p=='+1': return f(c+7,s+1,'+7')+f(c*4,s+1,'*4')
        if p=='+7': return f(c+1,s+1,'+1')+f(c*4,s+1,'*4')
        if p=='*4': return f(c+1,s+1,'+1')+f(c+7,s+1,'+7')
        if p=='': return f(c+1,s+1,'+1')+f(c+7,s+1,'+7')+f(c*4,s+1,'*4')

print(f(1,0,''))
print(len(d))
```

Ответ: 1091793

Telegram: @fast_ege

DZ_232_prog_13

Задача № 13 (4274)

Исполнитель Калькулятор преобразует число, записанное на экране. У исполнителя есть три команды, которым присвоены номера:

1. Прибавь 1
2. Прибавь 3
3. Умножь на 2

Первая команда увеличивает число на экране на 1, вторая увеличивает его на 3, третья – умножает на 2. Программа для исполнителя – это последовательность команд. Сколько существует программ, которые преобразуют исходное число 2 в число 51, и при этом траектория вычислений содержит число 18 и не содержит число 33. Также программа не должна содержать двух команд «Умножь на 2» подряд

Ссылка на видео-разбор с таймингом: https://vk.com/video-205546952_456241281?t=0h29m30s

Решение

Что бы ускорить процесс, напишем from functools import *

@lru_cache(None).

Напишем рекурсивную функцию, которая пройдет различные числовые траектории из одного числа до конечного и посчитает их количество. Функция будет принимать четыре параметра:

- c: текущее число.
- e: конечное число, к которому нужно прийти.
- k18: счетчик, количество раз, которые траектория проходит через число 18.
- p: предыдущие выполненные команды.

1. Если текущее число c = 18, то k18 равно 1, то есть мы попали в число 18.
2. Если текущее число c больше целевого числа e, или c равно 33, то вернуть 0.
3. Если текущее число c равно целевому числу e, то проверяется, проходит ли траектория через число 18. Если шагов да, k18 = 1, возвращается 1 (это значит, что найдено одно решение). Если не проходит, возвращается 0, потому что это не удовлетворяет условиям задачи.
4. Если текущее число c меньше целевого числа e, и предыдущая команда не была «умножить на 2», то вычисляются возможные пути для трех действий:

· Увеличить c на 1,
· Увеличить c на 3
· Умножить c на 2.

Рекурсивно вызывается функция для каждого из этих новых значений числа, значения k18 и соответствующего значения p. Результаты всех этих вызовов суммируются.

5. Если текущее число c меньше целевого числа e, и предыдущая команда была «умножить на 2», то вычисляются возможные пути для двух действий:

· Увеличить c на 1,
· Увеличить c на 3

Рекурсивно вызывается функция для каждого из этих новых значений числа, значения k18 и соответствующего значения p. Результаты этих вызовов суммируются.

Выводим на экран результат работы функции при c=2, e=51, k18=0 и p = ‘ ’. То есть считаем количество траекторий из 2 в 51, при том что изначально у нас нет прохождения через число 18 и функция ранее не запускалась, последовательность p не содержит ни одной команды.

```
from functools import *

@lru_cache(None)
def f(c,e,k18,p):
    if c==18: k18 = 1
    if c>e or c==33: return 0
    if c==e: return k18==1
    if c<e and p!='*2': return f(c+1,e,k18,'+1')+f(c+3,e,k18,'+3')+\
       f(c*2,e,k18,'*2')
    if c<e and p=='*2': return f(c+1,e,k18,'+1')+f(c+3,e,k18,'+3')

print(f(2,51,0,''))
```

Ответ: 43413100

Telegram: @fast_ege