Permutations with multiples 03/02/2019

itertools is coded in C . It’s fast. over 500 times faster than my Rexx program

import itertools

str\_a = 'DAFFODIL1234'

>>> str\_a\_list = list(str\_a)

>>> print(str\_a\_list)

['D', 'A', 'F', 'F', 'O', 'D', 'I', 'L', '1', '2', '3', '4']

>>> len(set(itertools.permutations(baby, 2)))

7

>>> len(set(itertools.permutations(str\_a\_list, 2)))

92

>>> len(set(itertools.permutations(str\_a\_list, 3)))

774

>>> len(set(itertools.permutations(str\_a\_list, 4)))

5910

>>> len(set(itertools.permutations(str\_a\_list, 5)))

40560

>>> len(set(itertools.permutations(str\_a\_list, 6)))

246960

>>> len(set(itertools.permutations(str\_a\_list, 6)))

246960

>>> len(set(itertools.permutations(str\_a\_list, 6)))

246960

>>>

/home/webvieww/public\_html/cgi-bin/math/perm\_counter\_sub.py

no spacing happiness

adsfdsaf

Python results:

**The count of Permutations is   
214318359246270926756547708278831157432016020595379605444850831990413727203735660110186709968760375988053208408719190361945145344000000000000000000000  
for N= 100 and R= 99  
and counts for duplicates of: 5 6 7**

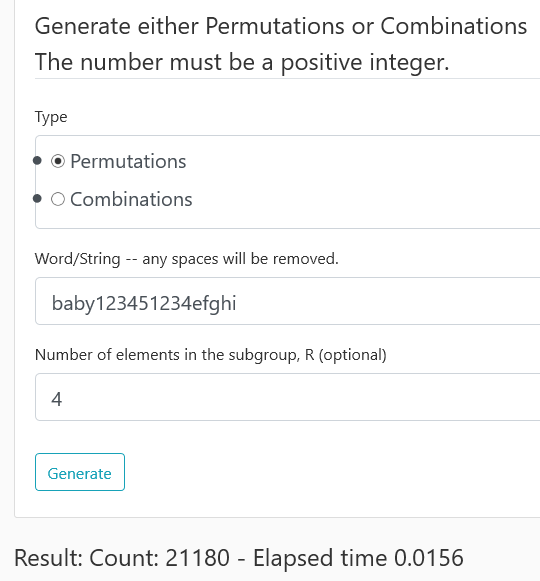
**The count of Combinations is   
85  
for N= 100 and R= 99  
and counts for duplicates of: 5 6 7**

**Rexx Results:**

**The count of Permutations is   
214318359246270926756547708278831157432016020595379605444850831990413727203735660110186709968760375988053208408719190361945145344000000000000000000000  
214,318,359,246,270,926,756,547,708,278,831,157,432,016,020,595,379,605,444,850,831,990,413,727,203,735,660,110,186,709,968,760,375,988,053,208,408,719,190,361,945,145,344,000,000,000,000,000,000,000  
2.1431835925 x 10 149  
for N= 100 and R= 99   
and counts for duplicates of: 5 6 7**

**The count of Combinations is   
85   
8.5 x 10 1  
for N= 100 and R= 99   
and counts for duplicates of: 5 6 7**

My system:



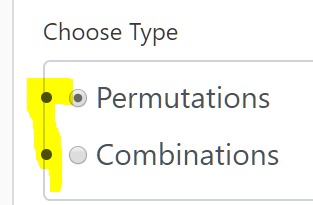
**GCP is similar**

**For Rexx, elapsed time has 9.1625**

**03/11/2019**

Generate 38,000 tuples Rexx 31.1 seconds, Python .014 seconds Python is **over 2,000 times faster**

These bullets don’t look good:



So add this code to html:

<!-- 03/11/2019 WVW: Remove the bullets since they are overlaying the box.

<style>

.nobullet {

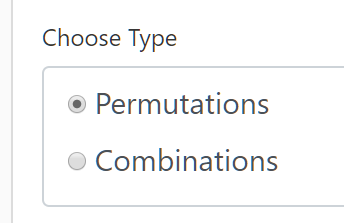
list-style-type:none;

}

</style>

{{ form.perm\_comb\_ind(class="form-control form-control-lg **nobullet**") }}

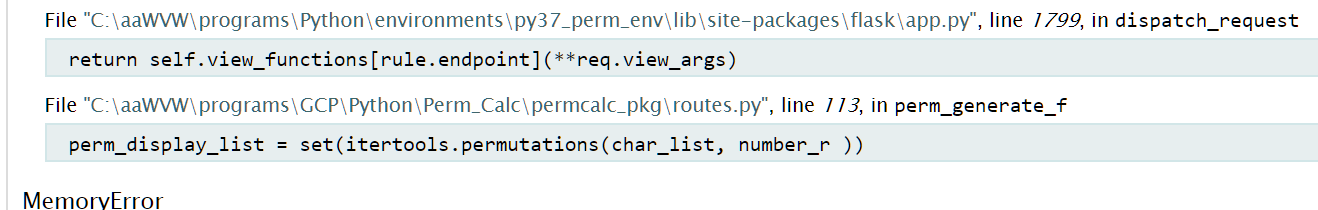
after change:



better

Generate for ‘warren van wyck’

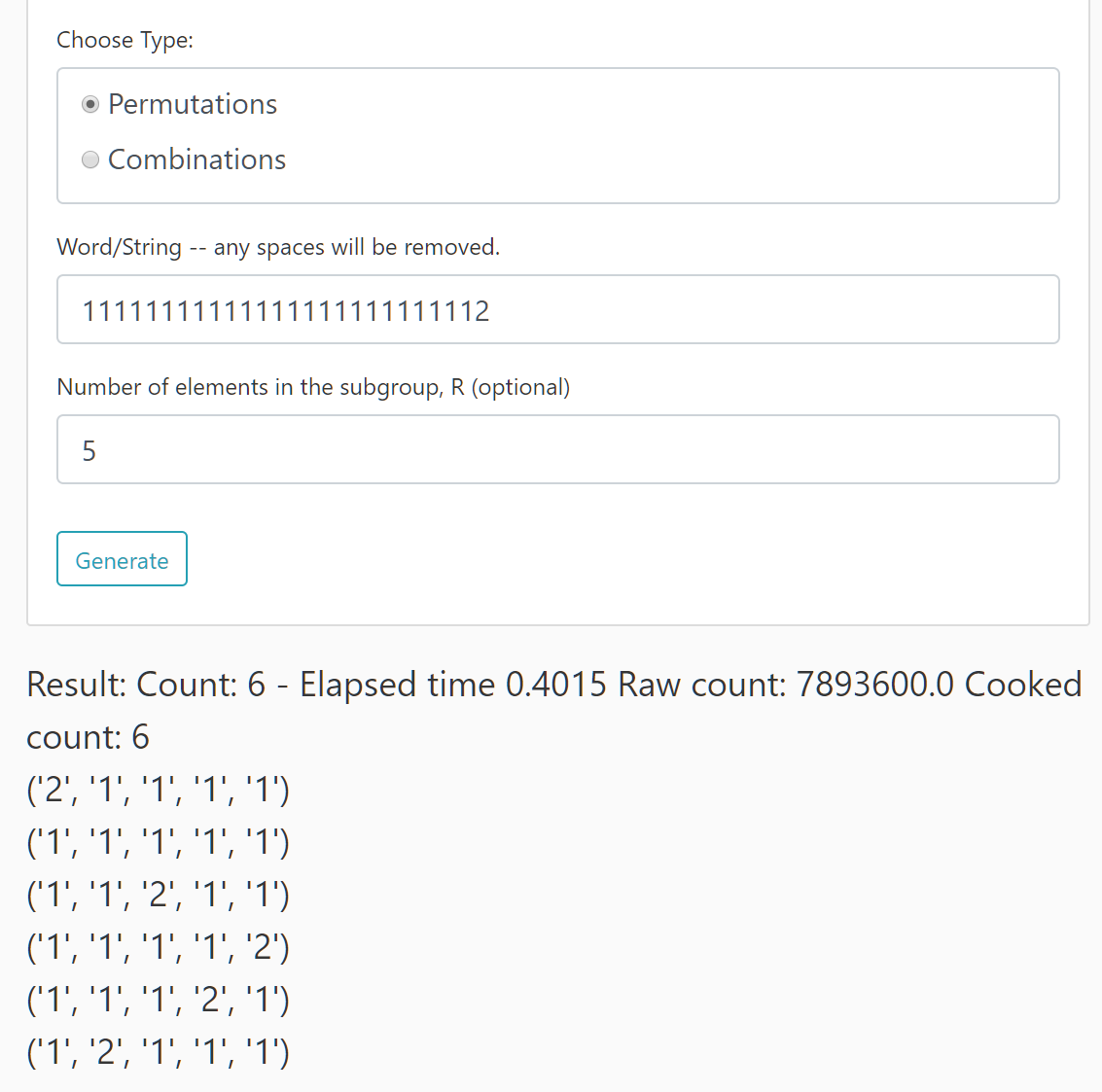
yields memory error:



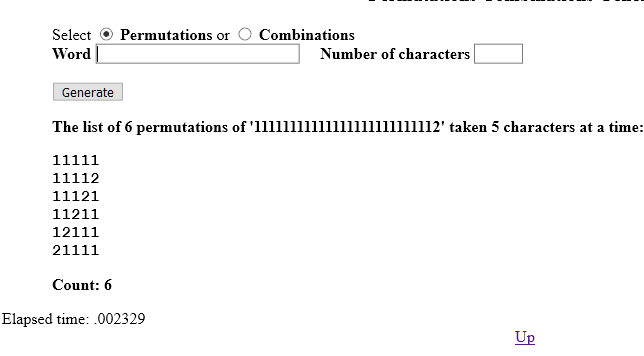
Result: 81,729,648,000 - Elapsed time 0.0

Result: 1,307,674,368,000

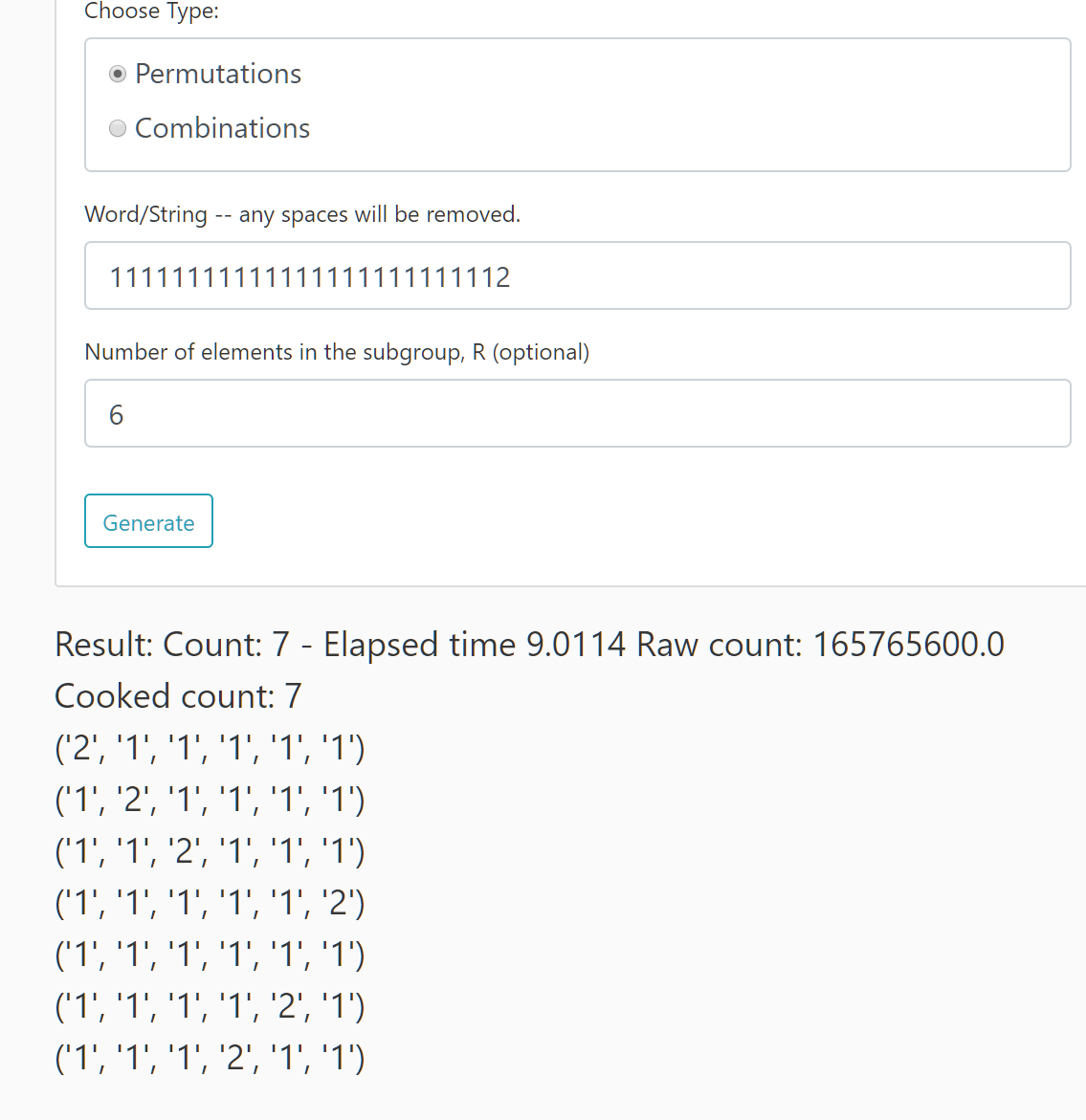
antidisestablishmentarianism

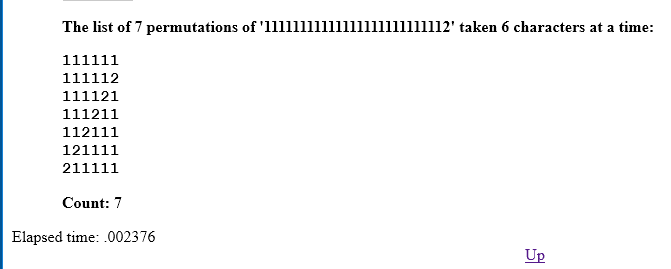


Rexx:



172 times faster





**3,792 times faster.**

**Now using**

sympy.utilities.iterables.**multiset\_permutations**(*m*, *size=None*, *g=None*)[[source]](https://docs.sympy.org/latest/_modules/sympy/utilities/iterables.html#multiset_permutations)

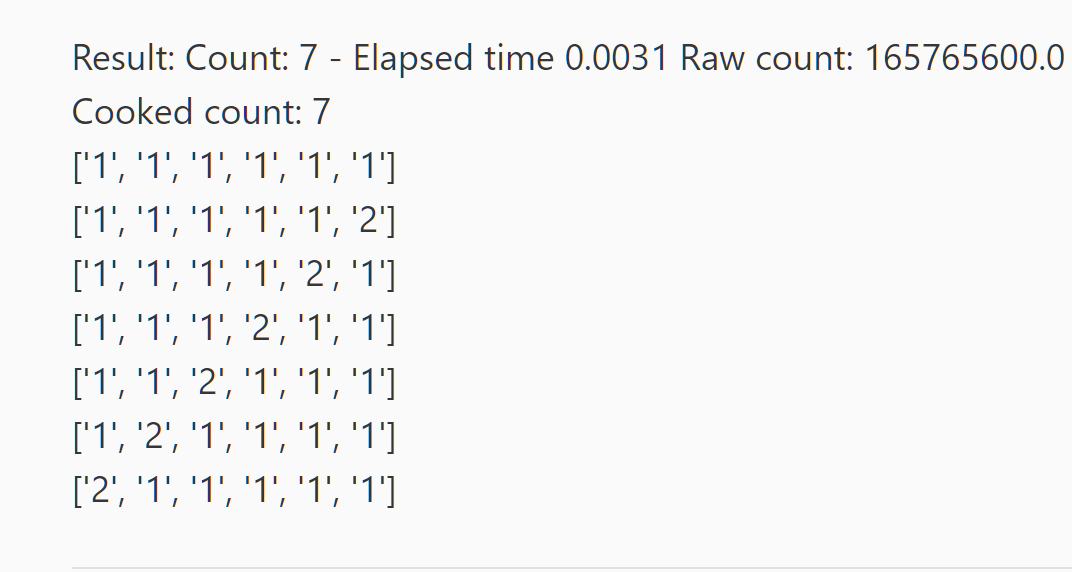
<https://docs.sympy.org/latest/modules/utilities/iterables.html?highlight=permutation#sympy.utilities.iterables.multiset_permutations>

**from sympy.utilities.iterables import multiset\_permutations**

**from sympy.utilities.iterables import multiset\_combinations**

**perm\_display\_gen = multiset\_permutations(char\_list, number\_r)**

**It’s superfast compared to itertools … about the same as Rexx program**



deploy to GCP

Result: Count: 7 - Elapsed time 0.0013 Raw count: 165765600.0 Cooked count: 7   
['1', '1', '1', '1', '1', '1']  
['1', '1', '1', '1', '1', '2']  
['1', '1', '1', '1', '2', '1']  
['1', '1', '1', '2', '1', '1']  
['1', '1', '2', '1', '1', '1']  
['1', '2', '1', '1', '1', '1']  
['2', '1', '1', '1', '1', '1']

from

https://docs.sympy.org/latest/\_modules/sympy/utilities/iterables.html

[[docs]](https://docs.sympy.org/latest/modules/utilities/iterables.html#sympy.utilities.iterables.multiset_permutations)**def** multiset\_permutations(m, size=**None**, g=**None**):

*"""*

*Return the unique permutations of multiset ``m``.*

*Examples*

*========*

*>>> from sympy.utilities.iterables import multiset\_permutations*

*>>> from sympy import factorial*

*>>> [''.join(i) for i in multiset\_permutations('aab')]*

*['aab', 'aba', 'baa']*

*>>> factorial(len('banana'))*

*720*

*>>> len(list(multiset\_permutations('banana')))*

*60*

*"""*

**if** g **is** **None**:

**if** type(m) **is** dict:

g = [[k, m[k]] **for** k **in** ordered(m)]

**else**:

m = list(ordered(m))

g = [list(i) **for** i **in** group(m, multiple=**False**)]

**del** m

do = [gi **for** gi **in** g **if** gi[1] > 0]

SUM = sum([gi[1] **for** gi **in** do])

**if** **not** do **or** size **is** **not** **None** **and** (size > SUM **or** size < 1):

**if** size < 1:

**yield** []

**return**

**elif** size == 1:

**for** k, v **in** do:

**yield** [k]

**elif** len(do) == 1:

k, v = do[0]

v = v **if** size **is** **None** **else** (size **if** size <= v **else** 0)

**yield** [k **for** i **in** range(v)]

**elif** all(v == 1 **for** k, v **in** do):

**for** p **in** permutations([k **for** k, v **in** do], size):

**yield** list(p)

**else**:

size = size **if** size **is** **not** **None** **else** SUM

**for** i, (k, v) **in** enumerate(do):

do[i][1] -= 1

**for** j **in** multiset\_permutations(**None**, size - 1, do):

**if** j:

**yield** [k] + j

do[i][1] += 1

>>> import numpy as np

>>> coef\_a = [1, 3, 2]

>>> coef\_b = [1, 0.5, 4, 3]

>>> pa =np.poly1d(coef\_a)

>>> pa

poly1d([1, 3, 2])

>>> pb =np.poly1d(coef\_b)

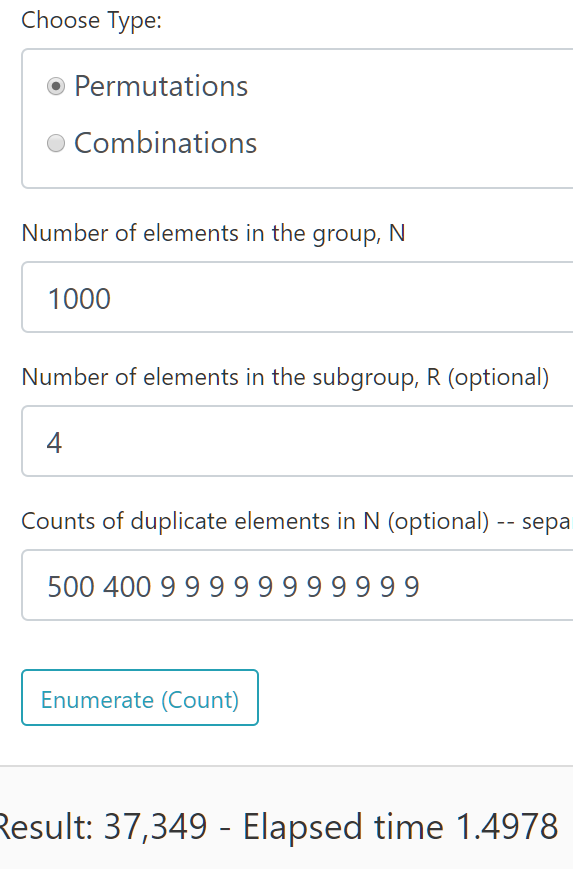
>>> pc = np.polymul(pa, pb)

>>> pc

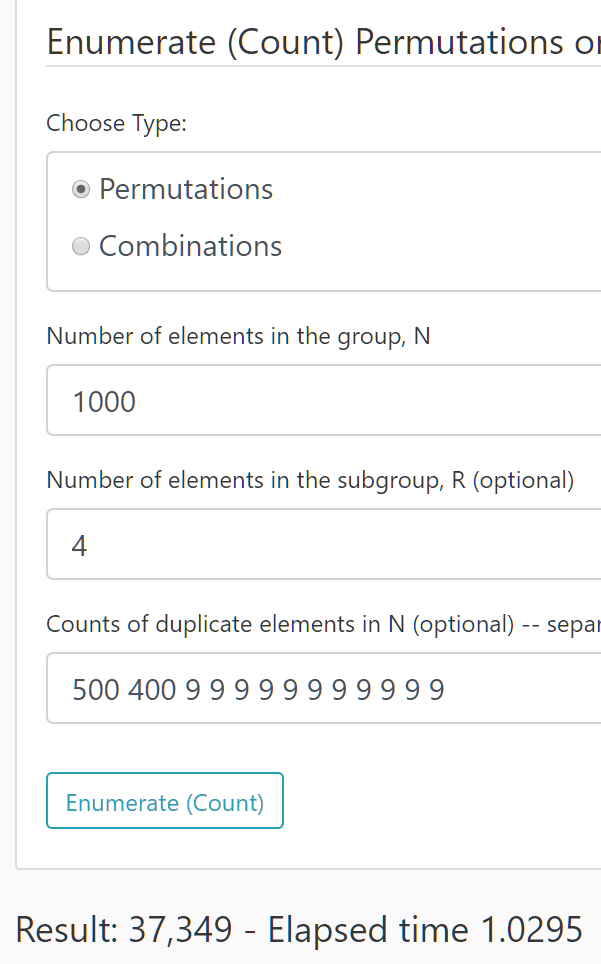
poly1d([ 1. , 3.5, 7.5, 16. , 17. , 6. ])

>>>

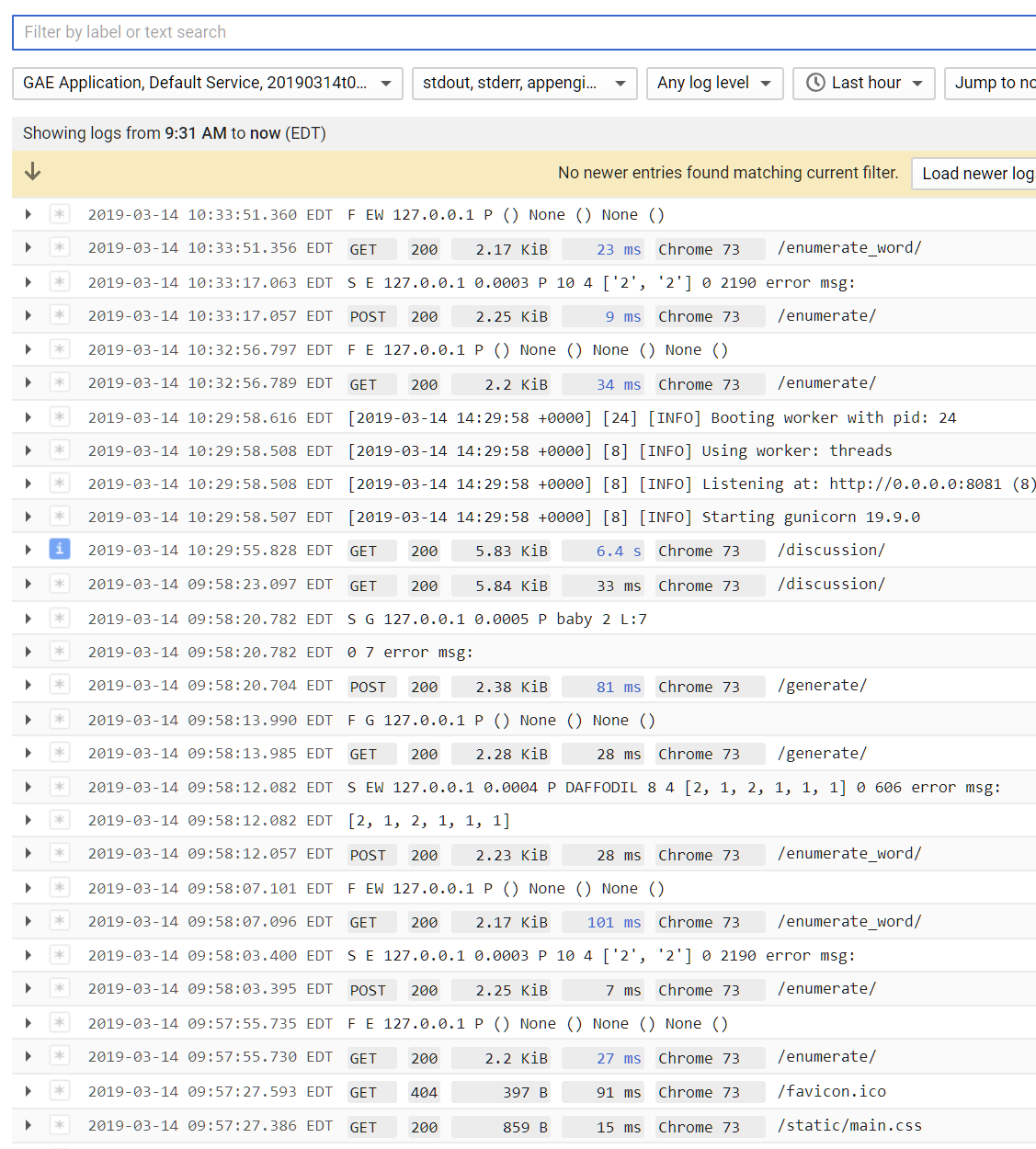
GCP:



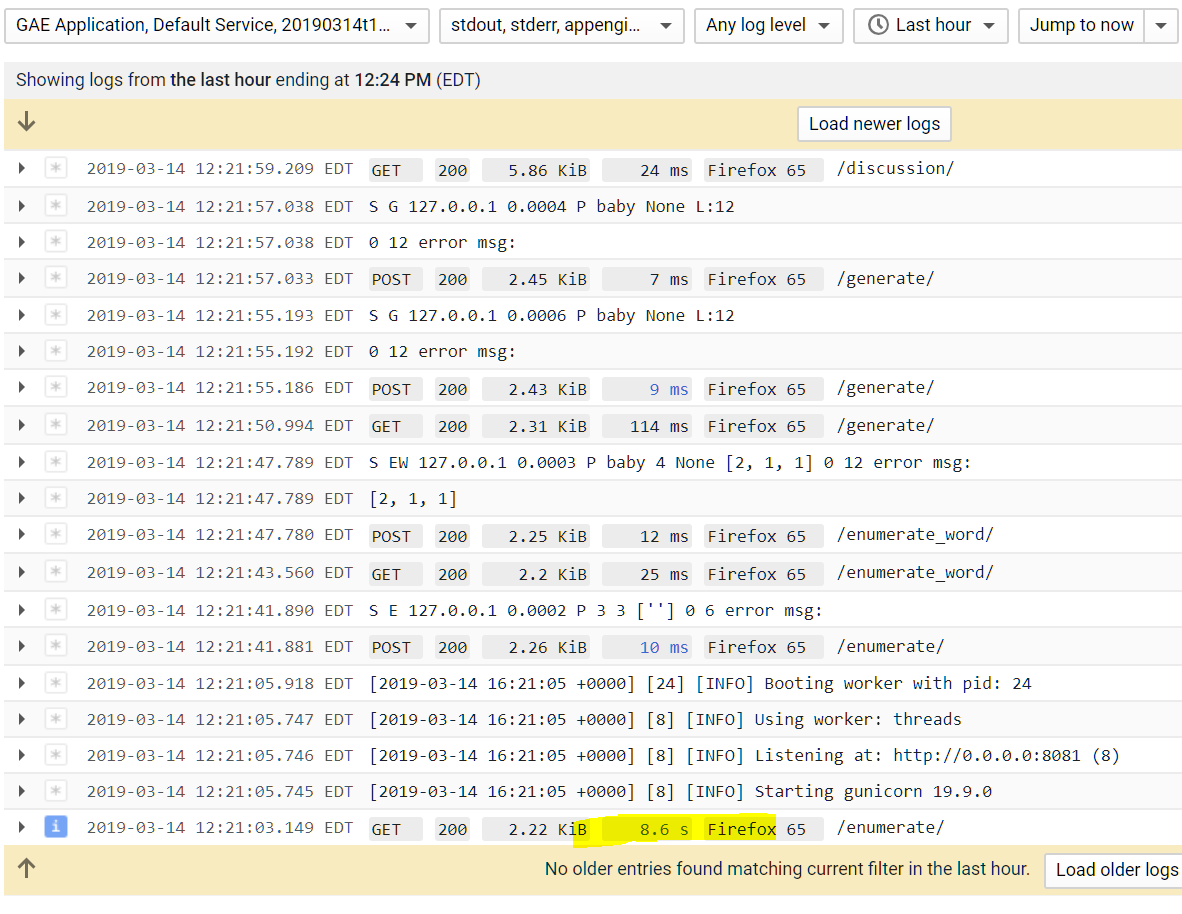
My PC:



Deploy GCP 03/14/2019



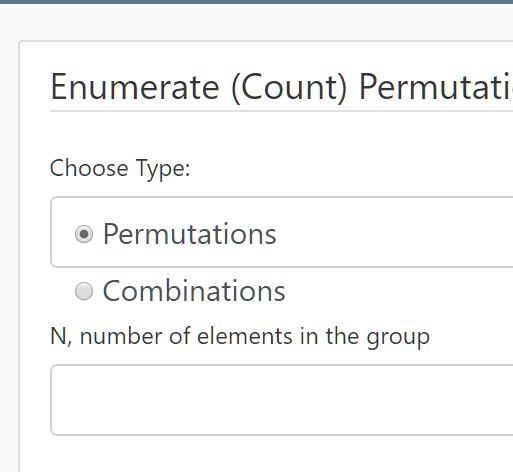
Note: The initial startup takes eight seconds



03/14/2019

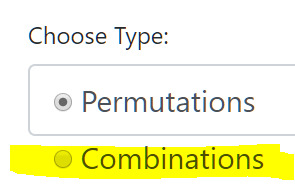
Problems with Bootstrip 4.3.1 NavBar is covering top to Main connect … not with 4.0.0 . Why???

With Boostrap 4.3.1 all browsers … not a problem with 4.0.0.

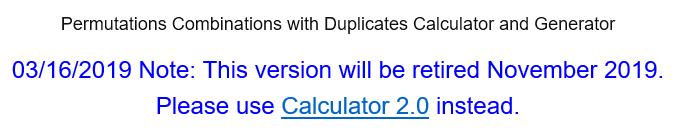


Learn more about Bootstrap … still a problems with 4.3.1

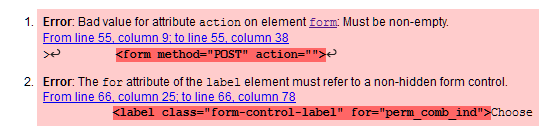
03/16/2019 11:50 am Deploy with 4.3.1 despite formatting issue



Add to index.htm On WebViewWorks at noon 3/16/19 Saturday



03/18/19 changes for Validation



03/25/2019

Restructure. Instead of the package ‘\_\_init\_\_.py’ implement, change to simpler module calls.

I restructured for AWS Lambda … I had issues with the package method.

