In [1]:

```
 #1 Einstein question
    #I found that I can solve this problem to use'constraint'
    #But I did't learn it in class, so I wonder how I can solve this que
```

In [24]:

```
house = ["blue", "green", "red", "white", "yellow"]
nationality = ["Brit", "Dane","German","Norwegian","Swede"]
beverage = ["beer", "coffee", "milk","tea","water"]
cigar = ["Blue Master", "Dunhill","Pall Mall", "Prince","Blend"]
pet = ["cat", "bird","dog","fish","horse"]

solution = house + nationality + beverage + cigar + pet


#dic1 = {'nationality':'Brit'}
#dic2 = {'nationality':'Dane'}
#dic3 = {'nationality':'German'}
#dic4 = {'nationality':'Norwegian'}
#dic5 = {'nationality':'Swede'}

#hints 1 to 15
hint1 = {"Brit","red"}
hint2 = {"Swded","dog"}
hint3 = {"dane","tea"}
#hint4. The green house is on the left of the white house (next to it).
hint5 = {"green", "coffee"}
hint6 = {"Pall Mall", "bird"}
hint7 = {"yellow", "Dunhill"}
#hint8. The man living in the house right in the center drinks milk.
#hint9. The Norwegian lives in the first house.
#hint10. The man who smokes blend lives next to the one who keeps cats.
#hint11. The man who keeps horses lives next to the man who smokes Dunhi
hint12 = {"Blue Master","beer"}
hint13 = {"German", "Prince"}
#hint14. The Norwegian lives next to the blue house.
#hint15. The man who smokes blend has a neighbor who drinks water.
```

In [15]:

```
  File "<ipython-input-15-60800007b42d>", line 157
    print ("Smoke: %s" % solution["smoke%d" % Einstein riddle pythoni])
                                                     ^
SyntaxError: invalid syntax
```

In [ ]:

In [22]:

```
# Houses
# 1 2 3 4 5

from constraint import *
problem = Problem()

nationality = ["영국", "스웨덴", "독일", "노르웨이", "덴마크"]
pet = ["개", "고양이", "새", "말", "물고기"]
cigarette = ["던힐", "블렌드", "펠몰", "프린스", "블루매스터"]
colour = ["빨강", "초록", "노랑", "파랑", "하양"]
beverage = ["커피", "우유", "맥주", "생수", "차"]

criteria = nationality + pet + cigarette + colour + beverage
problem.addVariables(criteria,[1,2,3,4,5])

problem.addConstraint(AllDifferentConstraint(), nationality)
problem.addConstraint(AllDifferentConstraint(), pet)
problem.addConstraint(AllDifferentConstraint(), cigarette)
problem.addConstraint(AllDifferentConstraint(), colour)
problem.addConstraint(AllDifferentConstraint(), beverage)

problem.addConstraint(lambda e, r: e == r, ["영국","빨강"])#1
problem.addConstraint(lambda s, d: s == d, ("스웨덴","개"))#2
problem.addConstraint(lambda c, g: c == g, ("커피","초록"))#5
problem.addConstraint(lambda u, t: u == t, ("덴마크","차"))#3
problem.addConstraint(lambda g, i: g-i == 1, ("하양","초록"))#4
problem.addConstraint(lambda o, s: o == s, ("펠몰","새"))#6
problem.addConstraint(lambda k, y: k == y, ("던힐","노랑"))#7
problem.addConstraint(InSetConstraint([3]), ["우유"])#8
problem.addConstraint(InSetConstraint([1]), ["노르웨이"])#9
problem.addConstraint(lambda c, f: abs(c-f) == 1, ("블렌드","고양이"))#10
problem.addConstraint(lambda k, h: abs(k-h) == 1, ("던힐","말"))#11
problem.addConstraint(lambda l, o: l == o, ["블루매스터","맥주"])#12
problem.addConstraint(lambda j, p: j == p, ["독일","프린스"])#13
problem.addConstraint(lambda k, h: abs(k-h) == 1, ("노르웨이","파랑"))#14

solution = problem.getSolutions()[0]

for i in range(1,6):
  for x in solution:
    if solution[x] == i:
      print (str(i), x)
```

```
---------------------------------------------------------------------------
----
ModuleNotFoundError                       Traceback (most recent call l
ast)
<ipython-input-22-87c87b1413da> in <module>()
      2 # 1 2 3 4 5
      3
----> 4 from Constraint import *
      5 problem = Problem()
      6
```

```
ModuleNotFoundError: No module named 'Constraint'
```

In [25]:

```
from constraint import *
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-25-fda0da6d8e39> in <module>()
----> 1 from constraint import *

ModuleNotFoundError: No module named 'constraint'
```

In [61]:

```
#2


def mergeSort(alist):
#test#    print("Splitting ",alist)

#lenth==1 -> basecase
    if len(alist)>1:
        mid = len(alist)//2
#devide into two part and get sorted half
        lefthalf = alist[:mid]
        righthalf = alist[mid:]

        mergeSort(lefthalf)
        mergeSort(righthalf)


#merge
        i=0
        j=0
        k=0
        while i < len(lefthalf) and j < len(righthalf):
            if lefthalf[i] < righthalf[j]:
                alist[k]=lefthalf[i]
                i=i+1
            else:
                alist[k]=righthalf[j]
                j=j+1
            k=k+1

#when just right are left
        while i < len(lefthalf):
            alist[k]=lefthalf[i]
            i=i+1
            k=k+1
#when just left are left
        while j < len(righthalf):
            alist[k]=righthalf[j]
            j=j+1
            k=k+1
    print("Merging ",alist)
```

In [62]:

```

#Generate a list between one to ten thosand
i = 1
a = [i]
while i < 10000 :
    i = i+1
    a.append(i)



#Randomize their order

import random
random.shuffle(a)



mergeSort(a)
```

```
Merging  [4938]
Merging  [4938, 9430]
Merging  [4938, 4987, 9430]
Merging  [4938, 4987, 8597, 8598, 9430]
Merging  [7903]
Merging  [4457]
Merging  [4457, 7903]
Merging  [7358]
Merging  [7009]
Merging  [1591]
Merging  [1591, 7009]
Merging  [1591, 7009, 7358]
Merging  [1591, 4457, 7009, 7358, 7903]

Merging  [1591, 4457, 4938, 4987, 7009, 7358, 7903, 8597, 8598, 9430]
Merging  [5124]
Merging  [8879]
Merging  [5124, 8879]
Merging  [7363]
Merging  [2372]
Merging  [7307]
```

In [ ]:

In [ ]:

In [63]:

```
#3-------------------------------------------------------------------
```

In [86]:

```
#3
import string
fhand = open('mailbox.txt')
counts = 0
for line in fhand :
    #print(line)

    inp = fhand.readline()
    tem = inp.find('From')
    #print(tem)

    if tem == 0 :
        #print(inp)
        counts = counts+1
        words = inp.split()
        for word in words :
            if word.find('@')!= -1 :
                print(word)
                key = word.find('@') + 1
                answer = word[key:]
                print(answer)
                print("\n")

print("the total number of ines starting with From : ", counts)
```

```
stephen.marquard@uct.ac.za
uct.ac.za


louis@media.berkeley.edu
media.berkeley.edu


zqian@umich.edu
umich.edu


rjlowe@iupui.edu
iupui.edu


zqian@umich.edu
umich.edu


the total number of ines starting with From :  5
```

In [87]:

```
#4---------------------------------------------------------------
```

In [13]:

```
#4
#1. Open file
import string
fhand = open('ROMEO.txt')

#2. Create an empty 'new words' list
counts = []
#3. Create a loop of strings for each line
for line in fhand:
    line = line.lower()
    words = line.split()

    for word in words:
        if word not in counts:
            counts.append(word)


#7. When the loop is finished sort the new words list so that it is alph

#print(counts)
counts.sort()
print(counts)

print("the number of list elements : ", len(counts))

    #print(tem)

#2. Create an empty 'new words' list
#3. Create a loop of strings for each line
#4. For every new string create a list consisting of new words
#5. Check if every element in the list is in the 'new words' list.
#6. If it is not repeated append the new word to the list.
#7. When the loop is finished sort the new words list so that it is alph
#8. Print the list elements and the number of elements.
```

```
['already', 'and', 'arise', 'breaks', 'but', 'east', 'envious', 'fair',
'grief', 'is', 'it', 'juliet', 'kill', 'light', 'moon', 'pale', 'sick',
'soft', 'sun', 'the', 'through', 'what', 'who', 'window', 'with', 'yond
er']
the number of list elements :  26
```

In [14]:

```
#5-----------------------------------------------------------------------
```

In [15]:

```
#5

fname = input('Enter the file name: ')
try :
    fhand = open(fname)
except :
    print('File cannot be opened:', fname)
    exit()

counts = dict()

for line in fhand:
    words = line.split()
    # print(words)
    for word in words :
        counts[word] = counts.get(word,0) + 1
print(counts)
```

```
Enter the file name: ROMEO.txt
{'But': 1, 'soft': 1, 'what': 1, 'light': 1, 'through': 1, 'yonder': 1,
'window': 1, 'breaks': 1, 'It': 1, 'is': 3, 'the': 3, 'east': 1, 'and':
3, 'Juliet': 1, 'sun': 2, 'Arise': 1, 'fair': 1, 'kill': 1, 'envious':
1, 'moon': 1, 'Who': 1, 'already': 1, 'sick': 1, 'pale': 1, 'with': 1,
'grief': 1}
```

In [ ]: