HA_2_report

January 25, 2018

1 Task 1

```
In [94]: def sumThreeFive(n):
             summa = 0
             for i in range(1,n):
                 if (i % 3) == 0 or (i % 5) == 0:
                     summa+=i
             return summa
         sumThreeFive(10000)

Out[94]: 23331668
```

2 Task 2

```
In [674]: M = {1: 1, 2: 2}

          def fib(n):
              if n in M:
                  return M[n]
              M[n] = fib(n - 1) + fib(n - 2)
              return M[n]

In [678]: fib(200)

Out[678]: 453973694165307953197296969697410619233826
```

3 Task 3

```
In [ ]: import pandas as pd
        import numpy as np

        file_ = open('words-list-russian.txt', 'r')
        file_obj = file_.readlines()
        file_.close()
```

Write words in an array and get rid of them

```
In [ ]: words = []
        for word in file_obj:
            words.append(word.strip())
```

Sort letters in words to use them further as keys for anagramms

```
In [ ]: words_sort = []
        for word in words:
            words_sort.append(''.join(sorted(word)))
        words_sort_tup = tuple(words_sort)

        word_keys = set(words_sort)
```

Create dictionary and print the result for keys with more than 4 items

```
In [725]: dict_words = dict()
          for value in word_keys:
              temp = []
              for c, word in enumerate(words_sort_tup,0):
                  if value == word:
                      temp.append(words[c])
              if len(temp) > 3:
                  dict_words[value] = temp
          dict_words

Out[725]: {'': ['', '', '', ''],
           '': ['', '', '', ''],
           '': ['', '', '', ''],
           '': ['', '', '', '', '', ''],
           '': ['', '', '', ''],
           '': ['', '', '', ''],
           '': ['', '', '', '']}
```

4 Task 4

Loaded words get from the previos Task. They are in the variable 'words'. With a method .permutations() of library itertools we can get all permutations of letters (different size and order) from word 'lekarstvo', which we will write in an array 'cure_keys_list

```
In [ ]: import itertools

        cure_keys_list = []
        for c in range(3,len('')+1):
            for i in itertools.permutations('',c):
                cure_keys_list.append(''.join(i))
```

Now we will use internal functions of set's comparing. The answer will be put in a variable 'ans'

```
In [181]: ans = set(cure_keys)&set(words)
          print(ans, len(ans))
```

```
{'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '
```

5 Task 5

n is a number of rounds (words) in a game. Initialize empty arrays.

```
In [ ]: import random

        n = 10
        five_letters = []
        game_words = []
        game_letters = []
```

Choose 5-letters words

```
In [ ]: for c, value in enumerate(set(words)):
            if len(value) == 5:
                five_letters.append(value)
```

Build list of size n

```
In [ ]: for i in range(0,n):
            c = random.randint(0,2069)
            game_words.append(five_letters[c])
```

Split words for games in letters

```
In [ ]: for i in range(0,10):
            game_letters.append(list(game_words[i]))
```

We will ask user in infinite loop

```
In [636]: for i in game_letters:
              while 1:
                  guess = input("Enter the word:")
                  letters = list(guess)
                  if letters != i:
                      equal_letters = set(letters) & set(i)
                      print("You guessed %d" %len(equal_letters))
                  else:
                      print("You win")
                      break
```

```
['', '', '', '', '', '', '', '', '', '']
Enter the word:
You guessed 2
Enter the word:
You win
Enter the word:
You guessed 2
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
Enter the word:
You win
```

6 Task 6

Here we will just guess words from set of words in a random order. Algorithm DOESN'T USE number of guesses letters.

```
In [ ]: for c, value in enumerate(set(words)):
            if len(value) == 5:
                five_letters.append(value)
        five_symbols = []
        for i in range(0,len(five_letters)):
            five_symbols.append(list(five_letters[i]))
        print(" ")
        while 1:
            n = random.randint(0,len(five_letters)-1)
            print(five_letters[n])
            symbol = input()
            if symbol == '!':
                print('You win')
            else:
                five_letters.pop(n)
```

```
2

0

2

2

3

4

5

6

7
```

7 Task 7

```
In [574]: import re
          import urllib.request
          from bs4 import BeautifulSoup

          url = 'http://www.belstat.gov.by/ofitsialnaya-statistika/makroekonomika-i-okruzhayus
          html = urllib.request.urlopen(url).read()
          soup = BeautifulSoup(html, 'html.parser') #class object creation
```

Let's find all data that is referred to table with tag 'p'

```
In [575]: part_1 = soup.find_all('p')
```

Let's find in found data only russian words and numbers with a regular expression. Put in a new array data without empty values

```
In [576]: c=[]
          for i in part_1:
              c.append(re.findall(r'\d+|[--]+',str(i)))
          c_without_empty = [x for x in c if x]
```

From table on the web-site we see that the last data from the table is 9 993 so we will delete all items after this one

```
In [ ]: last_data = c_without_empty.index(['9','993'])
        number = len(c_without_empty) - last_data - 1
        for i in range(0, number):
            c_without_empty.pop()
```

Also we will delete strings with dates

```
In [577]: c_without_empty.pop(0)
          c_without_empty.pop(0)

Out[577]: ['24', '01', '2018']
```

Initialize some empty arrays for final table content

```
In [ ]: head = []
        lines = []
        values = []
        counter = 0
```

Let's get out names of columns and at the same time delete them from the list. There are 8 columns that's why loop till 8

```
In [ ]: while counter < 8:
            head.append(c_without_empty[0])
            c_without_empty.pop(0)
            counter += 1
        counter = 0
```

Names of lines have the biggest lenght -> by this way we determine their location in an array and get them out

```
In [ ]: while counter < len(c_without_empty):
            if len(c_without_empty[counter])>2:
                line = ''
                for j in c_without_empty[counter]:
                    line += j + ' '
                lines.append(line)
                c_without_empty.pop(counter)
                counter = counter -1
            counter += 1
```

Insert empty value

```
In [578]: c_without_empty.insert(8, '')
```

Let's combine nimbers via ','

```
In [ ]: for c,i in enumerate(c_without_empty):
            line = ''
            for j in c_without_empty[c]:
                line += j + ','
            values.append(line)
```

Let's add empty values for the first line

```
In [579]: counter = 0
          while counter < 8:
              values.insert(0, '')
              counter+=1
```

Let's make reshape because vector-column and vector-line is not the same in python

```
In [580]: x = np.reshape(values, (4, 8))
          head = np.reshape(head, (1,8))
```

Create new object DataFrame and show the table with his help

```
In [581]: df = pd.DataFrame(x,index=list(lines),columns=list(head))
          df
```

```
Out[581]:                                                      2009        2010   \


               2016                 142,091,   170,466,
                 ...                  107,7,
                ...    14,946,    17,962,

                                                               2011        2012   \


               2016                 307,245,   547,617,
                 ...     105,5,     101,7,
                ...    32,433,    57,860,

                                                               2013        2014   \


               2016                 670,688,   805,793,
                 ...     101,0,     101,7,
                ...    70,852,    85,048,

                                                               2015      2016


               2016                 899,098,   94,949,
                 ...      96,2,     97,5,
                ...    94,745,    9,993,
```