# CUT-UP METHOD PYTHON-IMPLEMENTATION

## IDENTIFICATION OF PROBLEM:

Implementation of the Bowie, Burroughs, Gysin & Permutated Prose cut-up method in python application: Cut-up method involves breaking down the given text into many pieces (columns or cuts) and rearranging them in a specific given order of columns of users choice or random order, this will produce a whole new dimension of the given text, which entirely differs in sense of the original given input data. We have to do this using python by getting input data and remaking it as a cut-up text, there are many methods by which we can implement this in python.

## PROBLEM ANALYSIS:

1. The input data should be given to the python application via a input file which contains the input data or it can be given directly during the run-time and stored in a variable.
2. As per the user input the given text should be cut into multiple columns (A, B, C, D, E, F and etc.) ensuring the equality of the split columns and stored in an array.
3. In the given order of arrangement the columns should be rearranged and stored in an array.
4. There are two logics to combine the spliced columns. If Input is “ACB” directly join the text in each column. If input is “A+BC”, write the whole “A” column first and combine the rest of the columns.
5. The rearranged columns should be joined to make a new and different text that is the cut-up text.
6. The output -file should be opened in writing mode and the output data which is generated by the program should be written in the output file.

## POTENTIAL SOLUTIONS:

1. We can just get input data to find number of words and split them equally as per the input number of columns , now the columns are cut and stored in normal list ,we have to shuffle them in the order specified by the user that is something like “A+C+B”.
2. Using the array and numpy[numerical python] library in python to implement the cut-up method , the columns can be stored in the numpy n-dimensional array and they can be rearranged as per the input requirement of user.
3. Using Dictionary in python to store the key value pairs of the columns and their respective data and the text can easily be accessed in the order needed by passing the key(column) to the dictionary variable.

## APPROACH AND PSEUDOCODE:

The second way using numpy array library seems to be fast and tidy, in numpy nd-array we can split and store the columns of the given text by iteration and we can rearrange the array by using reshape() function by passing the parameters to join the columns and produce the resulted cut-up text.

### Base algorithm:

1. Define the column-dictionary and read data from the input file and store it.

1. Get input from the user for order of columns and iterate the lines in the given text and split the paragraph into specified number of columns.
2. Close the input file and store the split columns in .
3. Permutate the column as per the input given by the user and store them in a numpy nd-array and then reshape the numpy array using reshape() - ndarray.reshape (no.of.lines,no.of.column).
4. Open the output file with write permission and iterate over the reshaped columns in the array and write the columns as per the order given as input columns in the output file.
5. Close the output file after writing the output.

### Pseudocode:

IMPORT array ,numpy

DEFINE FUNCTION main():

SET coldict TO {"A": 1,"B": 2,"C": 3,"D": 4,"E": 5,"F": 6,}

SET myfile TO open(r"input\_file", 'rt')

SET Exp TO str(INPUT('Enter Input Columns')).upper()

SET noofcol TO 0

FOR i IN Exp:

IF i != '+':

noofcol += 1

SET noi ,col1, col2 TO []

SET lines TO myfile.readlines()

SET l TO 0

SET counter TO 1

#iterate lines IN paragraph and split the paragraph

FOR line IN lines:

SET # noi TO []

IF counter EQUALS 1:

SET l TO len(line)

SET strpos TO int(l / noofcol)

SET val TO 0

FOR i IN range(0, noofcol - 1):

val += strpos

noi.append(val)

noi.append(l)

counter += 1

SET start TO 0

SET itr=1

FOR i IN noi:

temp=line[start:i]

IF temp==' ':

temp=temp[1:i-1]

col1.append(temp)

#col1.append(line[start:i])

SET start TO i

itr+=1

SET totrec=counter\*noofcol

SET diff=totrec-len(col1)

FOR i IN range(0,diff):

col1.append('')

myfile.close()

#Store the values IN array

SET arr1=array(col1)

SET arr2=arr1.reshape(counter,noofcol)

#OUTPUT(arr2)

SET vol TO 0

SET nos TO []

SET nos TO Exp.split('+')

# permutate the column as per INPUT and write it IN output text file

SET f TO open(r"op.txt", 'w+')

SET inc=1

FOR a IN nos:

SET col2=[]

SET noofcol=0

FOR k IN a:

col2.append(thisdict[k])

INCREMENT noofcol += 1

SET vol TO 0

FOR i IN range(0, counter):

SET op TO ''

b=0

FOR j IN col2:

b+=1

j= j - 1

SET op TO " ".join([op, arr2[i][j]])

IF b EQUALS noofcol and op !='':

f.write(op)

OUTPUT(op)

IF inc<len(nos):

f.write('\n')

OUTPUT(op)

inc+=1

f.close()

main()

## PYTHON CODE-IMPLEMENTATION:

import array as arr

from numpy import \*

def main():

    #Created Dictionary for taking column from given input.

    thisdict = {

        "A": 1,

        "B": 2,

        "C": 3,

        "D": 4,

        "E": 5,

        "F": 6,

    }

    #open text file which has paragraph

    myfile = open(r"input.txt", 'rt')

    #Getting input

    Exp = str(input('Enter the input'))

    Exp=Exp.upper()

    noofcol = 0

    #Find Number of columns to split

    for i in Exp:

        if i != '+':

            noofcol += 1

    noi = []

    col1 = []

    col2 = []

    lines = myfile.readlines()

    l = 0

    counter = 1

    #iterate lines in paragraph and split the paragraph

    for line in lines:

        # noi = []

        if counter == 1:

            l = len(line)

            strpos = int(l / noofcol)

            val = 0

            for i in range(0, noofcol - 1):

                val += strpos

                noi.append(val)

            noi.append(l)

        counter += 1

        start = 0

        itr=1

        for i in noi:

            temp=line[start:i]

            if temp==' ':

                temp=temp[1:i-1]

            col1.append(temp)

            #col1.append(line[start:i])

            start = i

            itr+=1

        totrec=counter\*noofcol

        diff=totrec-len(col1)

        for i in range(0,diff):

            col1.append('')

    myfile.close()

    #Store the values in array

    arr1=array(col1)

    arr2=arr1.reshape(counter,noofcol)

    #print(arr2)

    vol = 0

    nos = []

    nos = Exp.split('+')

    # permutate the column as per input and write it in output  text file

    f = open(r"op.txt", 'w+')

    inc=1

    for a in nos:

        col2=[]

        noofcol=0

        for k in a:

            col2.append(thisdict[k])

            noofcol += 1

        vol = 0

        for i in range(0, counter):

            op = ''

            b=0

            for j in col2:

                b+=1

                j= j - 1

                op = " ".join([op, arr2[i][j]])

                if b == noofcol and op !='':

                    f.write(op)

                    print(op)

                    if inc<len(nos):

                        f.write('\n')

                        print(op)

        inc+=1

    f.close()

main()

## EVALUATION:

This works fairly for any of the given input text it converts the given input into many columns as per the given input, and rearranges the split columns into a whole new text which is the cut-up text for the given input. There are some minor issues with white space and blanks in the output , yet the code runs perfectly and produces a good result.

#### INPUT:

Alice opened the door and found that it led into a small passage, not much larger than a rat-hole: she knelt down and looked along the passage into the loveliest garden you ever saw. How she longed to get out of that dark hall, and wander about among those beds of bright flowers and those cool fountains, but she could not even get her head through the doorway; 'and even if my head would go through,' thought poor Alice, 'it would be of very little use without my shoulders. Oh, how I wish I could shut up like a telescope! I think I could, if I only know how to begin.' For, you see, so many out-of-the-way things had happened lately, that Alice had begun to think that very few things indeed were really impossible.

#### OUTPUT:

The output of the program: when the input via the cli(command line interface) is given as “BCAFDE”

ooked along the passage into the loveliest garden you ever saw. How she longed to get out of that dark hall, and wander about among those beds of bright flowers and those cool fountains, but she could not even get her head through the doo Alice opened the door and found that it led into a small passage, not much larger than a rat-hole: she knelt down and l out-of-the-way things had happened lately, that Alice had begun to think that very few things indeed were really impossible. rway; 'and even if my head would go through,' thought poor Alice, 'it would be of very little use without my shoulders. Oh, how I wish I could shut up like a telescope! I think I could, if I only know how to begin.' For, you see, so many

## PROBLEMS AND ISSUES:

1. **Problem-**While iterating over the lines in the given text to split the paragraph there remained some of the un-split text at the end of the input data i.e, the text in the para is split into equal parts for the columns where as at last there is some words left.

**Solution-** The remaining words while cutting the last column should be added with the last column itself because there is only few words left un-split there is not a big difference in adding them to the last column.

1. **Issue-** While reshaping the numpy nd-array I was having some issues and then I keenly watched the algorithm and I reshaped the array by passing the(no.of.columns,no.of.lines) at first which gave a logical error.

**Solution-** then after looking the documentation of the numpy and applying the logic it should be array.reshape(no.of.lines,no.of.columns) and the code runs fine after solving this problems.

## DEMONSTRATION:

## TEST-CASE:

## Input.txt:-Alice in Wonderland a paragraph from chapter 1

Suddenly she came upon a little three-legged table, all made of solid glass; there was nothing on it except a tiny golden key, and Alice's first thought was that it might belong to one of the doors of the hall; but, alas! either the locks were too large, or the key was too small, but at any rate it would not open any of them. However, on the second time round, she came upon a low curtain she had not noticed before, and behind it was a little door about fifteen inches high: she tried the little golden key in the lock, and to her great delight it fitted!

## Runtime column order input:-”EBAFDC”

### Output.txt:-

upon a low curtain she had not noticed before, and behind it was a little door about fifteen g on it except a tiny golden key, and Alice's first thought was that it might belong to one o Suddenly she came upon a little three-legged table, all made of solid glass; there was nothin inches high: she tried the little golden key in the lock, and to her great delight it fitted! , but at any rate it would not open any of them. However, on the second time round, she came f the doors of the hall; but, alas! either the locks were too large, or the key was too small

#### Observation:

We can observe that the input from the given input text file is cut up into six columns(A,B,C,D,E,F) and rearranged in the order of the given input(E,B,A,F,D,C)

and written in the output text file.

## Python Libraries used:

### NumPy(numerical python):

NumPy is a general-purpose range-processing python package. The ndarray (NumPy Array) is a multidimensional array used to store values of same datatype. These arrays are indexed just like Sequences, starts with zero.It provides a high performance multidimensional array object and device to work with these arrays.

This is the basic package for scientific computing with Python. It has various features.

Some of the uses of NumPy are:

Powerful n-dimensional array object

C / C ++ and Fortran code integration tools

Useful Linear Algebra, Fourier Transforms and Random Number Capabilities

In addition to its obvious scientific uses, NumPy can also be used as an efficient multidimensional container of common data.

#### NumPy Nd-array(n-dimensional array):

The ndarray (NumPy Array) is a multidimensional array used to store values of same datatype it is a multidimensional in the sense it can have many dimensions of the data stored in a same variable.

#### Functions-used :- Ndarray.reshape(rows,columns)

Ndarray.reshape(rows,columns)- is used to reshape the ndarray into the different shape.the parameter passed in are number of rows and columns are in other words the shape of the new ndarray.