Aayush Shah
D1 - 19BCE245
19 March 2021

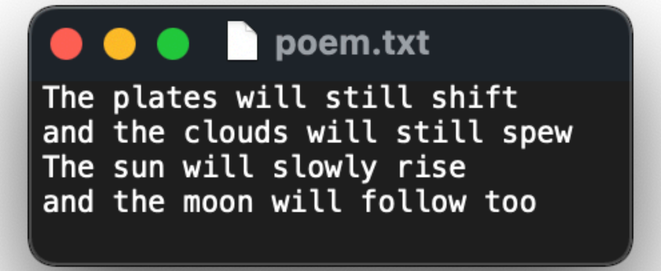
Practical 3

A. Write a python program that reads the contents of the file poem.txt and count the number of alphabets blank spaces lowercase letters and uppercase letters the number of words starting from vowel and the number of occurrences of each word in the file.

Code :

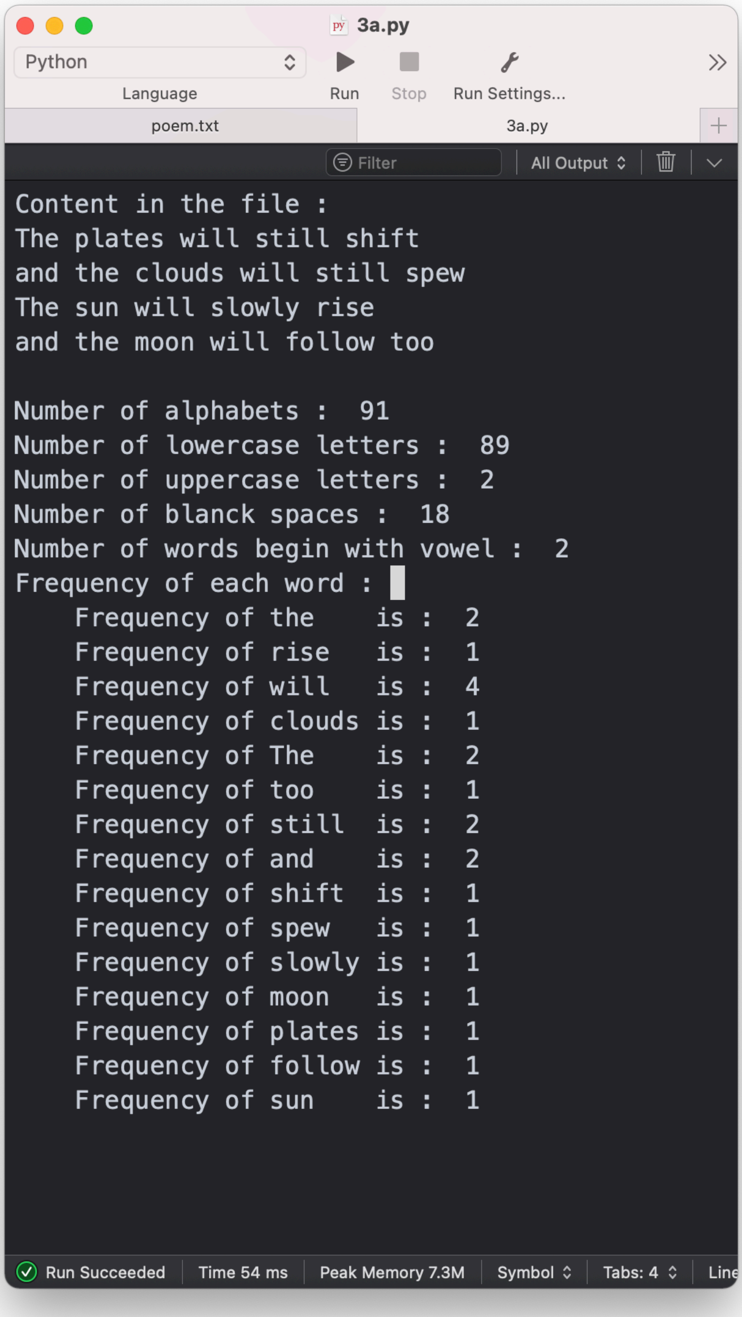
```python
poem_file = open("poem.txt") #by default opens in read mode.
content = poem_file.read()

alphabets = 0
blank_spaces = 0
lowercase = 0
uppercase = 0
words_starting_from_vowels = 0

for i in range(len(content)):
    if(content[i].isalpha()):
        alphabets += 1
        if(content[i].islower()):
            lowercase += 1
        if(content[i].isupper()):
            uppercase += 1
    elif(content[i] == ' '):
        blank_spaces += 1
    if(i==0 or content[i] == ' ' or content[i] == '\n'):
        if(i+1<len(content)):
            if(content[i+1]) in ['a','e','i','o','u','A','E','I','O','U']:
                words_starting_from_vowels += 1

        else:
            continue

print("Content in the file : ")
print(content,"\n")

print("Number of alphabets : ",alphabets)
print("Number of lowercase letters : ",lowercase)
print("Number of uppercase letters : ",uppercase)
print("Number of blanck spaces : ",blank_spaces)
print("Number of words begin with vowel : ",words_starting_from_vowels)

words = content.split()
unique_words = set(words)

print("Frequency of each word : ")
for word in unique_words:
  if(word != "\n" and word != " "):
       print("\tFrequency of",word.ljust(6),"is : “,words.count(word))
```

poem.txt file :



```
The plates will still shift
and the clouds will still spew
The sun will slowly rise
and the moon will follow too
```

Output :



```
Content in the file :
The plates will still shift
and the clouds will still spew
The sun will slowly rise
and the moon will follow too

Number of alphabets :  91
Number of lowercase letters :  89
Number of uppercase letters :  2
Number of blanck spaces :  18
Number of words begin with vowel :  2
Frequency of each word :
    Frequency of the    is :  2
    Frequency of rise   is :  1
    Frequency of will   is :  4
    Frequency of clouds is :  1
    Frequency of The    is :  2
    Frequency of too    is :  1
    Frequency of still  is :  2
    Frequency of and    is :  2
    Frequency of shift  is :  1
    Frequency of spew   is :  1
    Frequency of slowly is :  1
    Frequency of moon   is :  1
    Frequency of plates is :  1
    Frequency of follow is :  1
    Frequency of sun    is :  1
```

B. An organization wants to compute monthly wages to be paid to an employee in an organization. The input data is provided in two different files. File1 contains permanent employee data about employees (i.e. Empid, name, hourly wages), and File2 contains working hours information of each employee in the current month (i.e., empid and hours). Individual elements of data are separated by commas. Design a python program that reads both the files, computes the monthly wages of each employee and store in another file. Take both file names as command line arguments and check the respected exceptions for the same.

additional tasks by ma'am :

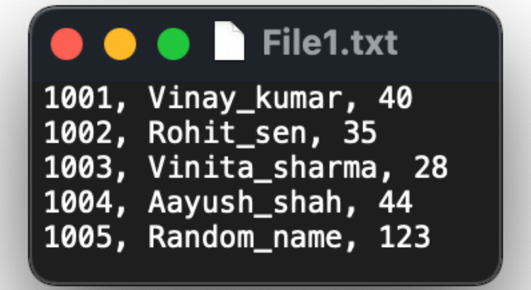
- *NA in case of there is no data is available.*
- *giving input file name in command line arguments as File1.txt and File2.txt*

Code :

```
import sys
try : #exception handling
    file1 = open(sys.argv[1])          #By default in read mode.
except IOError:
    print("Cannot find file.\nReading data from default file...")
    file1 = open("File1.txt")
flag = 1
matrix1 = []      #2D array for storing file1's data.
matrix2 = []      #2D array for storing file1's data.
data = ""    #var for storing each word
row1 = 0     #counter for rows of file1
row2 = 0     #counter for rows of file2
col = 0
mat_list = []          #1D array for storing each line's data.

while 1:
    #reading file character by character.
    char = file1.read(1)     #reading one char at a time
    if(char == ',' or char == '\n'):    #current data ended indicator
        mat_list.append(data)         #appending data to line's data
        data = ""      #erasing current word
        col += 1       #going to new data
        if(col==3):         #current line ended indicator [for increasing row value].
            col = 0         #going to new data in new line
            row1 += 1 #going to new row
            matrix1.append(mat_list)      #appending line's data in the matrix
            mat_list = []       #erasing line's data as it is  stored in matrix now.
    elif(char != ' '):       #current data is still being read
        data += char         #collecting chars for current data
    if not char:         #if  end  of line reached then exit the while loop
        break

#printing data
print("Data extracted from file 1 : ")
for i in range(row1):
    print("\t",end="")
    for j in range(3):
        print(matrix1[i][j],end="  ")
    print()

file1.close()          #closing file1


try : #exception handling
    file2 = open(sys.argv[2])          #By default in read mode.
except IOError:
    print("Cannot find file.\nReading data from default file...")
    file2 = open("File2.txt")

#reading file2
while 1:
    #reading file character by character.
    char = file2.read(1)      #reading one char at a time
    if(char == ',' or char == '\n'):    #current data ended indicator
        if(char=='\n' and data == ""):
            mat_list.append("NA")         #if data is not available then simply writing NA in that field
        else:
            mat_list.append(data)         #appending data to line's data
        data = ""       #erasing current word
        col += 1        #going to new data
        if(col==2):          #current line ended indicator [for increasing row value].
            col = 0          #going to new data in new line
            row2 += 1 #going to new row
            matrix2.append(mat_list)      #appending line's data in the matrix
            mat_list = []        #erasing line's data as it is  stored in matrix now.
    elif(char != ' '):       #current data is still being read
        data += char         #collecting chars for current data
    if not char:         #if  end  of line reached then exit the while loop
        if(col==1 and data == ""):
            mat_list.append("NA")
        else:
            mat_list.append(data)
        matrix2.append(mat_list)
        row2 += 1
        break

#printing data
print("\nData extracted from file 2 :  ")
for i in range(row2):
    print("\t",end="")
    for j in range(2):
        print(matrix2[i][j],end="  ")
    print()

file2.close()         #closing file2


file3 = open("File3.txt",'w') #File for storing calculated wages
```

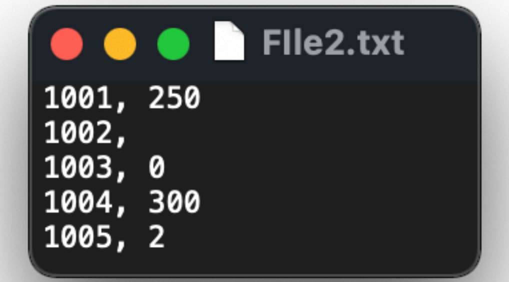
```
write_str = "ID".ljust(5) + "NAME".ljust(16) + "WAGES" + "\n" + "".ljust(30,"-") + "\n"  #writing heading in the file3

#printing salary along with writing it in File3.txt
print("\nCalculated monthly wages : ")
file3.write(write_str)
for i in range(row1):
    if(matrix2[i][1] == "NA"):
        print("\t",matrix1[i][0],matrix1[i][1].ljust(15),"NA")
        write_str = matrix1[i][0] + " " + matrix1[i][1].ljust(15) + " " + "NA" + "\n"
        file3.write(write_str)
    else:
        print("\t",matrix1[i][0],matrix1[i][1].ljust(15),int(matrix1[i][2])*int(matrix2[i][1]))
        write_str = matrix1[i][0] + " " + matrix1[i][1].ljust(15) + " " + str(int(matrix1[i][2])*int(matrix2[i][1])) + "\n"
        file3.write(write_str)

file3.close()

"""
Insert extra '\n' at the end of the file1.txt and do not insert any extra '\n' at the end of the file2.txt

command line arguments : File1.txt File2.txt
"""
```

File1.txt file :



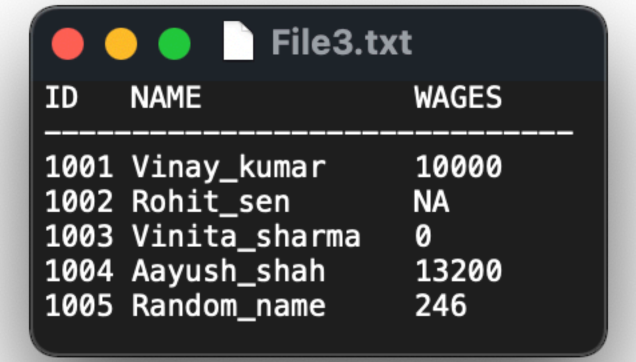
```
1001, Vinay_kumar, 40
1002, Rohit_sen, 35
1003, Vinita_sharma, 28
1004, Aayush_shah, 44
1005, Random_name, 123
```

File2.txt file :



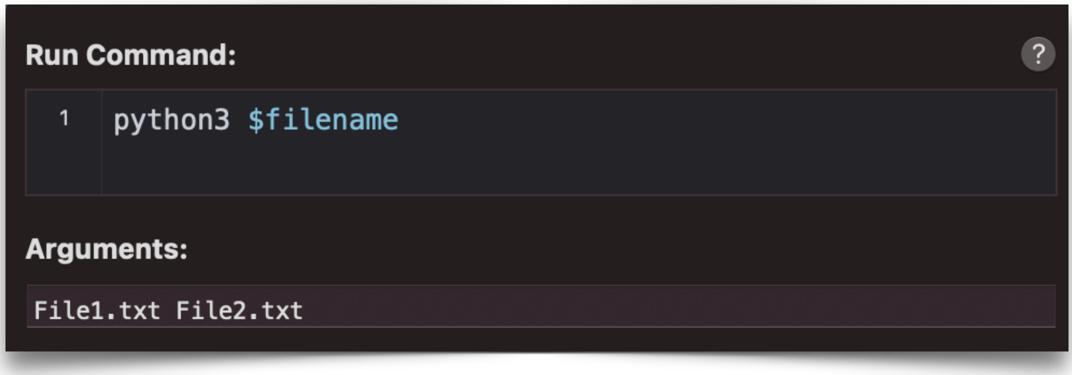
```
1001, 250
1002,
1003, 0
1004, 300
1005, 2
```

Generated File3.txt file :



```
ID   NAME             WAGES
------------------------------
1001 Vinay_kumar      10000
1002 Rohit_sen        NA
1003 Vinita_sharma    0
1004 Aayush_shah      13200
1005 Random_name      246
```

Output [given command line arguments and output window] :

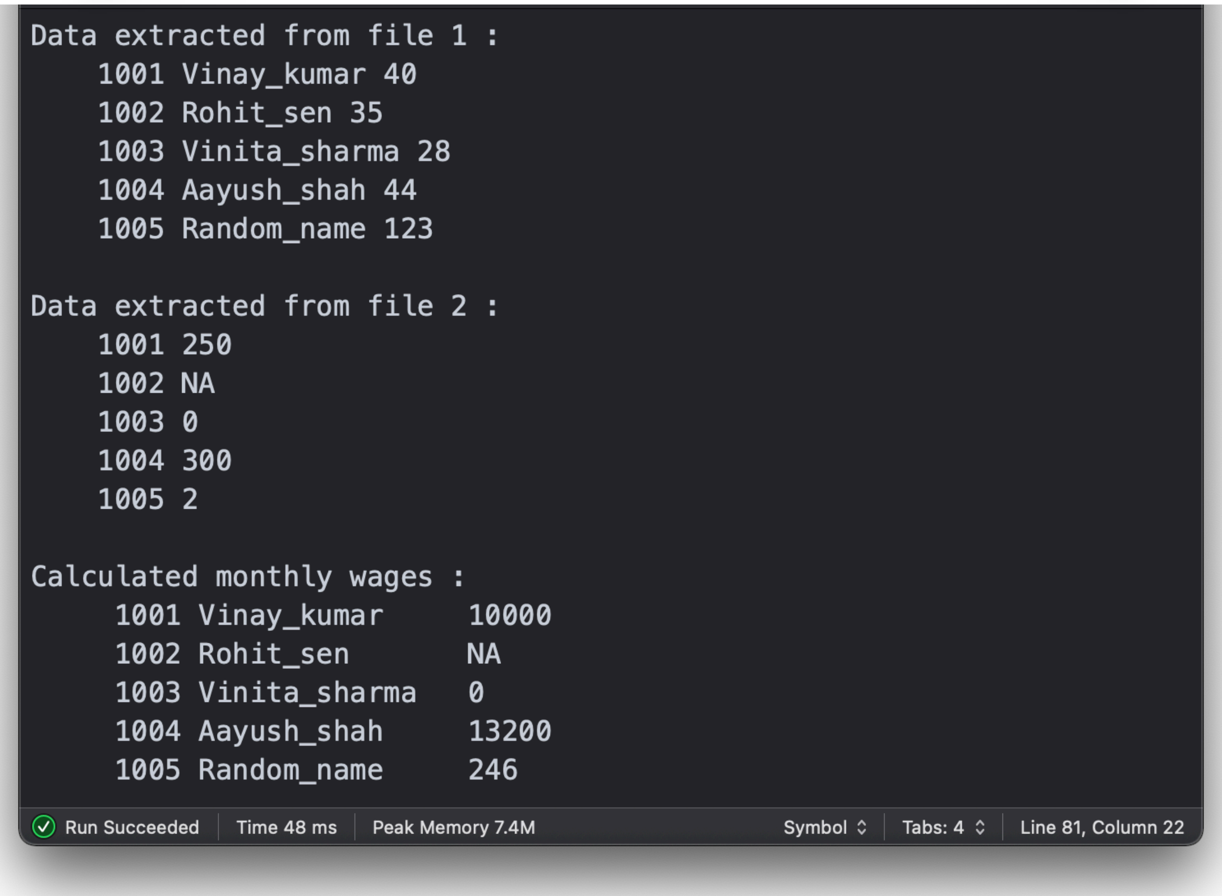


Run Command:

```
python3 $filename
```

Arguments:

```
File1.txt File2.txt
```



```
Data extracted from file 1 :
    1001 Vinay_kumar 40
    1002 Rohit_sen 35
    1003 Vinita_sharma 28
    1004 Aayush_shah 44
    1005 Random_name 123

Data extracted from file 2 :
    1001 250
    1002 NA
    1003 0
    1004 300
    1005 2

Calculated monthly wages :
     1001 Vinay_kumar     10000
     1002 Rohit_sen       NA
     1003 Vinita_sharma   0
     1004 Aayush_shah     13200
     1005 Random_name     246
```

C. Consider the following formula and evaluate the y value for the range of t values found in a file with format

additional tasks by ma'am :

- *Consider this formula : (1/2)*(initial_velocity)^2 *()*
- *giving input file name in command line arguments as File1.txt and File2.txt*

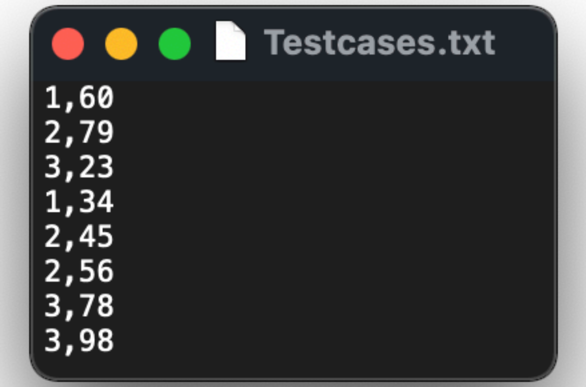
Code :

```
"""
testcases -> file which contains test cases          [format : id,velocity]
answers -> storing output       [format : id,velocity,breaking distance]
summary -> storing AVERAGE VELOCIY and TOTAL DISTANCE.
"""

from scipy.constants import g

def compute_d(initial_velocity,friction_coeff):
  return ((1/2)*(initial_velocity**2)/(friction_coeff*g))


testcases = open("Testcases.txt")    #opening testcases.txt in read mode
tests = []        #list for storing testcases.txt's data
test = []         #list for storing individual test
row = 0           #counter for rows in testcases
col = 0           #counter for columns in testcases [fixed to 2]
data = ""    #for storing each value in testcases

#reading data from testcases.txt.
while(1):
    char  = testcases.read(1)
# print(char,col,row,data,test,tests)
    if(char == ',' or char == '\n'):
        test.append(data)
        data = ""
        col += 1
        if(col == 2):
            col = 0
            row += 1
            tests.append(test)
            test = []
    elif(char != ' '):
        data += char
    if not char:
        test.append(data)
        tests.append(test)
        break

testcases.close()
#printing extracted data from testcases.txt
#print(tests)

answers = open("Answers.txt",'w')          #opening answers file for writing answers
friction_coeff = 0.3        #setting friction coefficient

final_tests = []       #storing data format : id,velocity,distance
final_test = []

#calculating answers.
print("Calculated breaking distances : ")
for i in range(len(tests)):
    final_test = []
    final_test.append(tests[i][0])
    final_test.append(tests[i][1])
    final_test.append(str(round(compute_d(int(tests[i][1]), friction_coeff),2)))
    final_tests.append(final_test)
    write_str = "ID :" + "  " + tests[i][0].ljust(10) + "  " + "VELOCITY :  " + "  " + tests[i][1].ljust(10) + "  " + "DISTANCE :  " + "  " + final_test[2] + "\n"
    print(write_str)
```

```
  answers.write(write_str)

#print(final_tests)


set_of_tests = {}          #stores in format => id : [total_vel,total_dis,count]
total_dis = 0
avg_vel = 0
count_vel = 0
for i in range(len(final_tests)):
# if set_of_tests.has_key(final_tests[i][0]) :
  if final_tests[i][0] in set_of_tests :
       set_of_tests[final_tests[i][0]][0] += int(final_tests[i][1]) #adding velocity
       set_of_tests[final_tests[i][0]][1] += float(final_tests[i][2]) #adding distance
       set_of_tests[final_tests[i][0]][2] += 1 #increasing counter
  else:
#      set_of_tests[final_tests[i][0]] = 0,0,1
       set_of_tests[final_tests[i][0]] = {0:int(final_tests[i][1]),1:float(final_tests[i][2]),2:1}

#print(set_of_tests)


summary = open("Summary.txt",'w')         #opening summary.txt in writing mode.
write_str = "ID".ljust(10) + "AVG-VELOCITY".ljust(15) + "TOTAL-DISTANCE\n" + "".ljust(40,"-") + "\n"
summary.write(write_str)        #defining heading.

#writing data
for test in set_of_tests:
  write_str = str(test).ljust(10) + str(round(set_of_tests[test][0]/set_of_tests[test][2],2)).ljust(15) + str(round(set_of_tests[test][1],2)) + "\n"
  summary.write(write_str)

  """
```

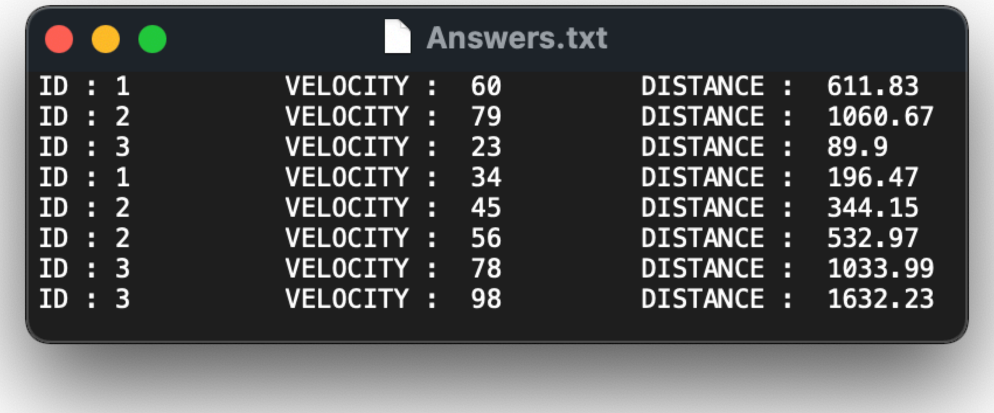
```
DO NOT ADD EXTRA '\n' at the end of the Testcases.txt
  file.
"""
```

Given Testcases.txt file :



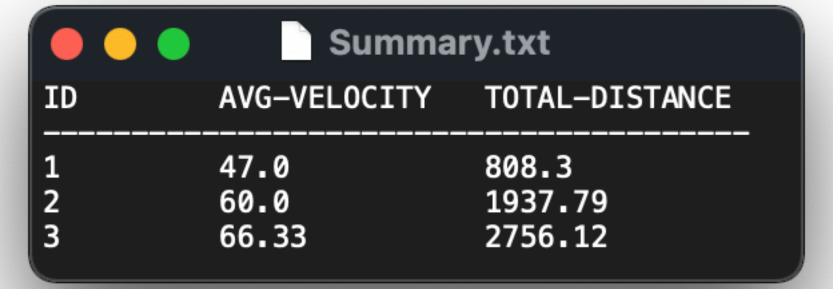
```
1,60
2,79
3,23
1,34
2,45
2,56
3,78
3,98
```

Generated Answers.txt file :



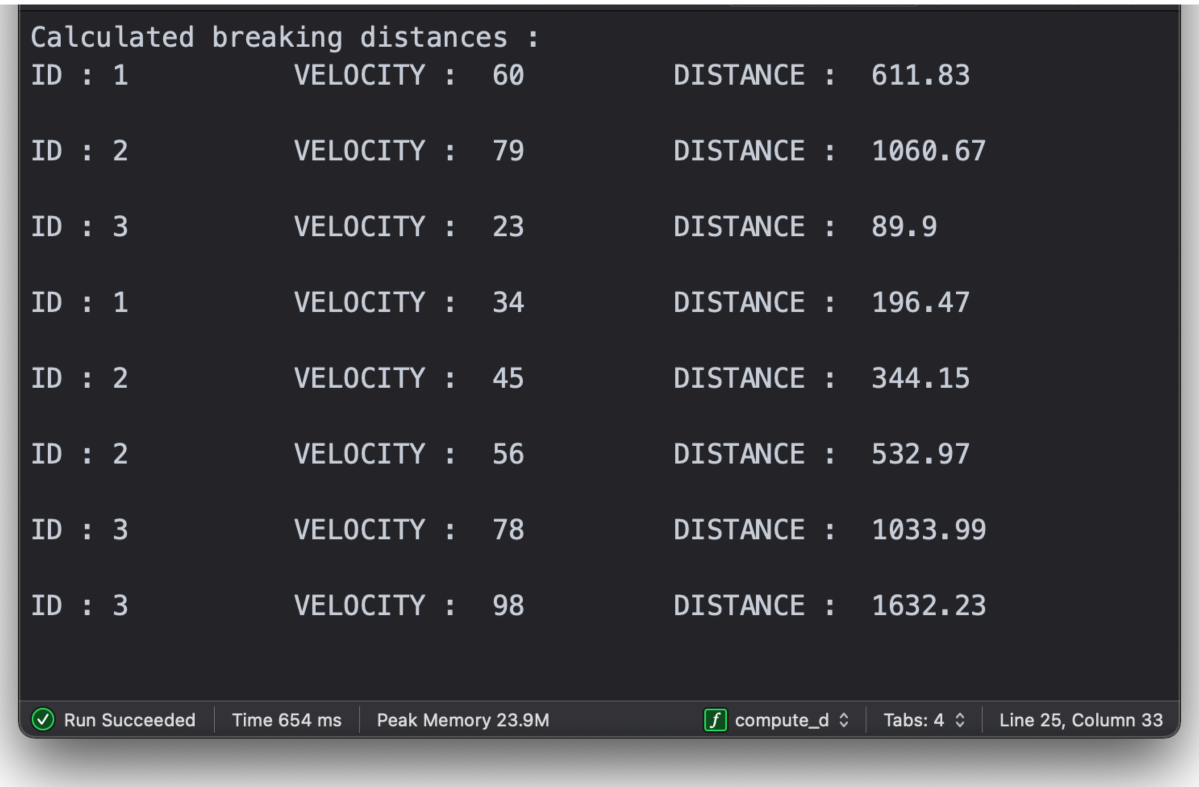
```
ID : 1          VELOCITY :  60          DISTANCE :  611.83
ID : 2          VELOCITY :  79          DISTANCE :  1060.67
ID : 3          VELOCITY :  23          DISTANCE :  89.9
ID : 1          VELOCITY :  34          DISTANCE :  196.47
ID : 2          VELOCITY :  45          DISTANCE :  344.15
ID : 2          VELOCITY :  56          DISTANCE :  532.97
ID : 3          VELOCITY :  78          DISTANCE :  1033.99
ID : 3          VELOCITY :  98          DISTANCE :  1632.23
```

Generated Summary.txt file :



```
ID          AVG-VELOCITY    TOTAL-DISTANCE
-----------------------------------------
1           47.0            808.3
2           60.0            1937.79
3           66.33           2756.12
```

Output :



```
Calculated breaking distances :
ID : 1          VELOCITY :  60         DISTANCE :  611.83

ID : 2          VELOCITY :  79         DISTANCE :  1060.67

ID : 3          VELOCITY :  23         DISTANCE :  89.9

ID : 1          VELOCITY :  34         DISTANCE :  196.47

ID : 2          VELOCITY :  45         DISTANCE :  344.15

ID : 2          VELOCITY :  56         DISTANCE :  532.97

ID : 3          VELOCITY :  78         DISTANCE :  1033.99

ID : 3          VELOCITY :  98         DISTANCE :  1632.23
```