Informatics Institute of Technology

Foundation program in higher education

Module: DOC 334 Introduction to Programming in Python – P2

Module Leader: Dr. Damitha Karunaratne

Module Co-Leader: Mr. Nishan Saliya

Assignment Number	:	01
Assignment Type	:	Individual
Issue Date	:	25th July 2020
Date of Submission	:	8th Auguet 2020 (On or before 10.00 AM)
Time	:	On or before 10.00 AM
Student ID	:	2019784
Student Name in Initials:		W.W.M.R.F
Student First Name	:	Rukshan
Student Surname	:	Fernando

Abstract

Python is a simple and amazing object-oriented programming language. It was initially made, harking back to the 1980's nevertheless observed it's first open release in 1991. After the release of Python 1.0 in 1994, it immediately got one of the favored programming languages for the creation of web applications in the Internet, alongside with Perl and PHP. It's creator, Guido van Rossum has had a significant influence in the Python advancement from it's first release and has a central role in choosing the direction of the Python improvement.

Python is regularly utilized as a scripting language for web applications in blend with the "mod python" module for the Apache web server. Python's easiness of utilization and capacity to integrate with various SDKs permits the formation of a wide range of programs for Windows, Linux, Mac OS and other operational systems.

Among the most significant characteristics of Python is the utilization of elegant syntax, which permits the clients to read program code effectively and which makes it appropriate for prototype development and distinctive ad-hoc programming task. Python accompanies an inbuilt development environment considered as IDLE and offers a huge standard library that underpins numerous basic programming assignments, for example, connecting to web servers, looking through text with ordinary expressions, reading and adjusting files. In interactive mode Python can without much of a stretch test even little portions of code.

Acknowledgement

I am honestly thankful to informatics Institute of Technology for giving this great subject and teaching us this Python programming language. This Individual Coursework was a huge impact to improve my Python programming skills and problem solving methods. I am also grateful to our Introduction to Programming 2 module lecture

Dr. Damitha Karunaratne and tutorial lecture Mr. Nishan Saliya for guiding and helping me in every stage of this subject. Without their guidance it would been very difficult for me to solve this question. I am also grateful to the Academic Skills for Higher Education module lectures Ms. Shyani_Siriwardene and Ms. Antoinette Hettiarachy for their guidance to make a professional and academic report. I hope this Assignment will help all those who interested in Python programming, to get some knowledge.

Contents

Abstract 2
Acknowledgement 3
Contents 4
List of figures 5
Introduction 6
Assignment Brief and Tasks to Complete 7
Assignment Brief 7
Tasks to Complete 8
Problem Statement 9
Solution Outline 10
Visual Representation Of The Assignment 11
Python Codes 12
1) perc.py 12
2) algorithm.py 14
3) sheet.py 18
4) html.py 20
Test Cases 22
Table Of Test Cases 22
Screenshot Of Test Cases 26
1) Test Case 01 (T01) 26
2) Test Case 02 (T02) 27
3) Test Case 03 (T03) 28
4) Test Case 04 (T04) 29
5) Test Case 05 (T05) 30
6) Test Case 06 (T06) 31
7) Test Case 07 (T07) 33
8) Test Case 08 (T08) 35
9) Test Case 09 (T09) 37
10) Test Case 10 (T10) 39

List of figures

Figure 1: Sample Percolator 7
Figure 2: Visual Representation Diagram 11
Figure 3: Test Case 01_Command Console 26
Figure 4: Test Case 02_Command Console 27
Figure 5: Test Case 03_Command Console 28
Figure 6: Test Case 03_Text File 28
Figure 7: Test Case 03_HTML File 28
Figure 8: Test Case 04_Command Console 29
Figure 9: Test Case 05_Command Console 30
Figure 10: Test Case 06_Command Console 31
Figure 11: Test Case 06_Text File 31
Figure 12: Test Case 06_HTML File 32
Figure 13: Test Case 07_Command Console 33
Figure 14: Test Case 07_Text File 33
Figure 15: Test Case 07_HTML File 34
Figure 16: Test Case 08_Command Console 35
Figure 17: Test Case 08_Text File 35
Figure 18: Test Case 08_HTML File 36
Figure 19: Test Case 09_Console Command 37
Figure 20: Test Case 09_Text File 37
Figure 21: Test Case 09_HTML File 38
Figure 22: Test Case 10_Console Command 39
Figure 23: Test Case 10_Text File 39
Figure 24: Test Case 10_HTML File 40

Introduction

Python is a widely utilized universally useful, high-level programming language. It was created by Guido van Rossum in 1991 and additionally developed by the Python Software Foundation. It was structured with an emphasis on code readability, and its syntax allows programmers to communicate their concepts in fewer lines of code. Python has a simple syntax like the English language. It runs on an interpreter system, meaning that the code can be executed when it is written. This means prototyping can be speedy. It can be treated in a procedural manner, an object-orientated way, or a functional way. Afterall, Python is a programming language that lets you work rapidly and integrate systems all the more proficiently. Specially Python is used for web development, software development, mathematics, system scripting.

Assignment Brief and Tasks to Complete

Assignment Brief

You are to create a Python program which will allow users to demonstrate a very simple percolation process.

Percolation is the process of a liquid slowly passing through a filter. This is how coffee is usually made. Your coursework is to create a program which mimics this concept.

A dynamic grid with 2 digits random numbers will be created. The grid will have some empty slots with no numbers, which again randomly generated. You program will check each column for possible percolations.

- The percolation is not possible for a column if the column consists of one or more empty spaces from top to bottom.
- The percolation is possible for a column if the entire column consists numbers from top to bottom You must state weather percolation of a column is possible or not at the end of each column.

C:\>perc 9x9

33	13	92	42	18	59	91	44	40
34	93	94	29	93	15	55	75	83
95	82	63	33	83	32	64	32	47
44		16	40	90		45	81	24
12	75	67	44	58	37	13	26	87
23	69	29	16		82	39	20	87
76	17	70		68	63	40	11	
24	82	10	53	32	88	14	23	80
53	94	92	59	24	39	13	70	63
OK	NO	OK	NO	NO	NO	OK	OK	NO

Figure 1: Sample Percolator

Tasks to Complete

1. You must use proper Python 3.x program constructs such as packages, modules, functions, variables, data structures, etc. to develop this program.
    - Hint : Remember the rubrics given for mini projects during tutorial sessions

2. The grid size is dynamically passed as a command line argument to the program. If no dimensions are passed, the default dimension for the grid is 5x5. The lowest dimensions must be 3x3 while the highest dimension must be 9x9. Some possible commands would be,
    - C:\>perc 3x4

        1. The above creates a 3x4 grid

    - C:\>perc

        1. The above creates a 5x5 default grid

3. Grid must be populated automatically with 2 digits random numbers

4. There will be some empty cells which are appearing randomly inside the grid

5. Display the status at the end of each column
    - Display OK if percolation is possible
    - Display NO if percolation is not possible

6. DO NOT use NumPy to generate the grid!
    - You will be awarded zero marks for using this or similar modules!
    - You need to create your own algorithm to generate this grid

7. You can use pretty table module if you like (say to create borders around numbers to enhance the grid appearance)

8. The resulting answer must be written to a text file so the result can be viewed later via notepad
    - Each result should go to a different text file

9. A challenge activity will be to generate the grid and save the result in a HTML file so you can see the answer in a web browser.
    - Hint: pretty table can be used for this
    - You'll get extra 10 marks for completing this task

Problem Statement

- In this course work its required to develop a python program to generate a simple percolation process. When the user input the size of the grid (Eg: 4x6) in command console, then it generate a dynamic grid with 2 numerals random numbers and sometimes it randomly generate empty space or spaces in the grid but there must be definitely only 1 space in a column.

- And also, user input size of the grid value should be equal or higher than 3x3 and equal or lower than 9x9. For an example user cannot input values like 2x2, 1x1, 2x3, 9x10, 10x12, …etc. Or else user can run the program with not entering a value, but then the program randomly generates the grid with default size of 5x5. Briefly we can summarize that arguments like,
    - Lowest size of the grid is 3x3
    - Highest size of the grid is 9x9
    - Default size of the grid is 5x5

- After that process next the program should be checking the columns of the grid. If there is any blank slot found in the grid, bottom of each column it displays a message as “NO”. If there is not any blank slot found in the grid, bottom of each column it displays a message as “OK”. Then display the dynamic gird.

- After all of that progress the dynamic grid result will be saved in the text file and then it converts as a HTML and saved it in a HTML file.

Solution Outline

- First of all, user should be entered a value for the size of the grid.

- In the program there are 3 type of commands,
  I. User entered size of the grid must be Higher than 3x3
  II. User entered size of the grid must be Lower than 9x9
  III. If User not entered size of the grid but he run the program, then size of the grid must be 5x5 (Default Dimension)

  If user entered size is follow these commands program will be continue. Or else if is not, user has to re-enter size of the grid.

- After then the next step is generate 2 numerals random number list. The numbers in the grid must be consists numbers. There can be empty spaces or not in the list, but there cannot be more than 1 empty space in a column. So for that we have to create functions.

- After that process our next progress is check if there any empty slot in a column, if is it we have to display the status of "NO". If there no empty slot in a column we have to display the status "OK". We have to execute this message ("OK" or "NO") on bottom of the every columns in the grid.

- Then the record of the grid print on the command prompt and generate into a text file in "record" folder.

- In the "algorithm" folder there is a text file called "algorithm.txt". Current file number is stored in that file.

- Then our final process is converting the text file in "record" folder to HTML and store it in "htmlcode" → "record" file direction.

- You can run this program again and again.

Visual Representation Of The Assignment

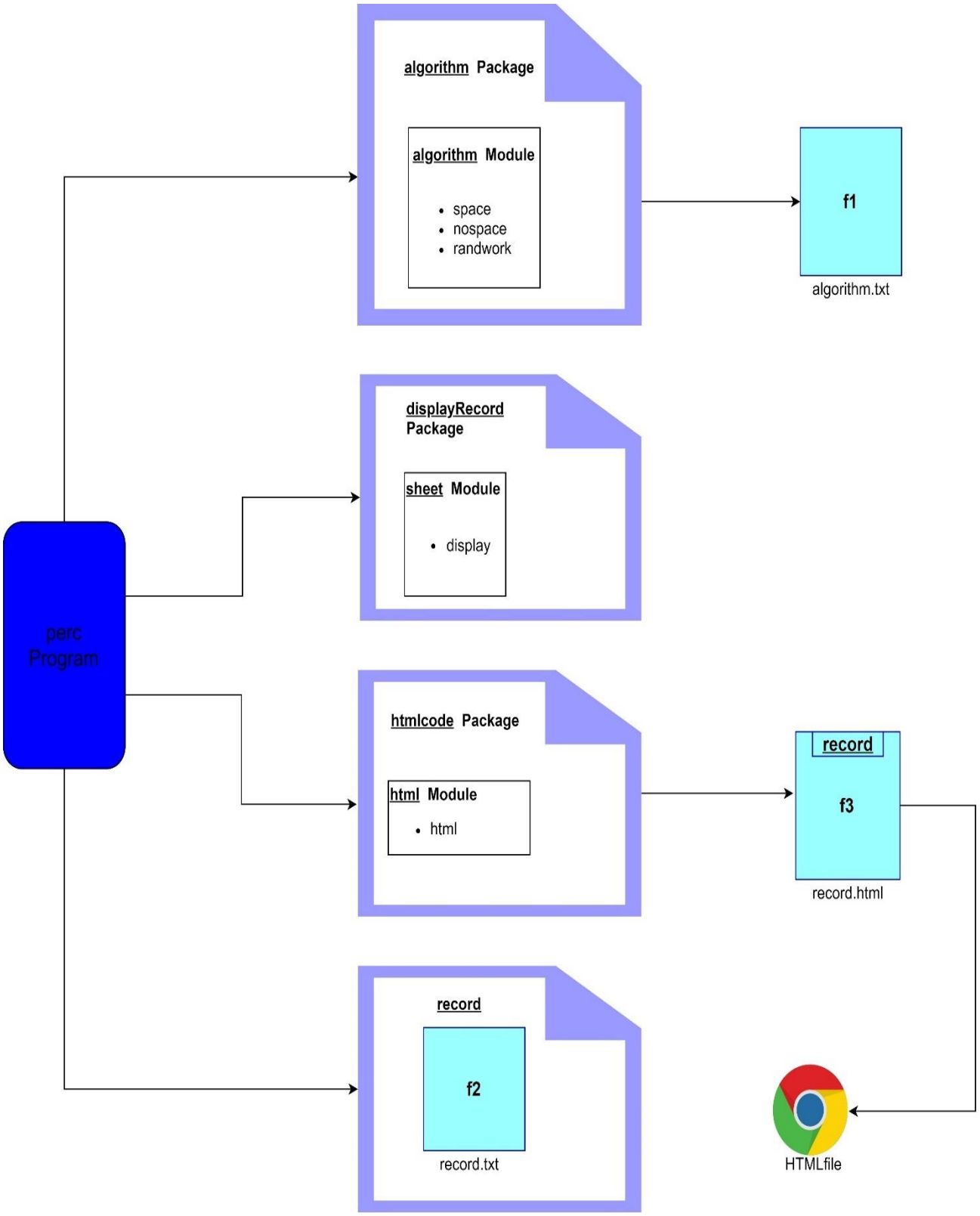


Figure 2: Visual Representation Diagram

Python Codes

1) perc.py

```
#----------------VARIABLES--------------------
  #grid              : argument controlling
  #value_1 and value_2: make use of for get argument first value and second values

from algorithm.algorithm import randwork
import sys

#discover the argument length
if len(sys.argv)==1:
   grid = " "

#obtain the last argument
else:
   grid = sys.argv[-1] #obtain the last argument

#create default value for the code
if grid == " " or grid == " ":
   value_1,value_2 = 5,5
   randwork(value_1,value_2)

#search for an error
else:
   if int(grid[0])>2 and int(grid[2])>2 and int(grid[0])<10 and int(grid[2])<10:
      value_2,value_1=int(grid[0]),int(grid[2])
```

```
        randwork(value_1,value_2)


#display the error message
    else:
        print("\n\t------Your Enter Grid Size is Valid------","\n")
        print("\t\t*Grid 3x3 should be the LOWEST DIMENSIONS for this system")
        print("\t\t*Grid 9x9 should be the HIGHEST DIMENSIONS for this system")
        print("\n\t------Please Correct the Grid Size and Try Again!------")
```

2) algorithm.py

```
#------------------VARIABLES-------------------
   # a,b      : using this variables as count in "for" loop
   # rntc     : to get the random numbers
   # spaceList: (10 - 100) list with given a space
   # numList  : (10 - 100) number list
   # table    : to get final value table
   # blank    : using for search a space in "rntc"
   # value_2  : coming from the perc.py file

import random
import sys
sys.path.insert(0,'displayRecord')
import sheet

def space(value_2,rntc):
#if 'rntc'(random number table column) have a space then this def works
#adding numbers to the table column (rntc)
#creating the final table in 'table'

   #creating the columns
   for a in range(value_2):

      #search spaces and stop data duplication
      if " " in rntc:
         rntc += (random.choice(numList) + ",")

      else:
```

```
        rntc += (random.choice(spaceList) + ",")



    #search column and insert "NO" or "OK"
    if a == value_2 - 1:
        if " " in rntc:
            rntc += ("NO,")

        else:
            rntc += ("OK,")

  table.append(rntc)

def nospace(value_2,rntc):
#if rntc(random numbers table column) have not a space then this def works
#adding numbers to random numbers table column(rntc)
#creating the final table(table)

  #creating the columns
  for a in range(value_2):

    #adding the numbers
    rntc += (random.choice(numList) + ",")

    #insert "NO" and "OK"
    if a == value_2 - 1:
      if " " in rntc:
        rntc += ("NO,")
```

```
        else:
            rntc += ("OK,")

    table.append(rntc)

def randwork(value_1,value_2):
#create 'spaceList' and 'numList' list
#search space in the random number table column 'rntc'
#stoping the space duplcation(again and again)

    global numList
    global spaceList
    global table

    table = []
    spaceList = [" ", ]
    numList = []
    blank = 0

    #create 'spaceList' and 'numList' lists
    for a in range(10, 100):
        spaceList.append(str(a))
        numList.append(str(a))

    #creating the table rows
    for b in range(value_1):

    #stoping the space duplcation(again and again)
```

```
    #asking the space def
    if blank == 0:
        rntc = ""
        space(value_2,rntc)
        blank = 1

    #asking the nospace def
    else:
        rntc = ""
        nospace(value_2,rntc)
        blank = 0

#print(table)
sheet.display(table)
```

3) sheet.py

```
#-----------------VARIABLES-----------------
   #f1      : algorithm.txt file in algorithm folder
   #f2      : create record file in record folder
   #spaceList: read the file and get the values
   #table    :data export from algorithm.py

def display(table):
#open algorithm.txt file in algorithm folder
#create the display file in record folder
#get table from algorithm.py and create the text file
#after then print the record

   #open algorithm.txt file or read only
   f1 = open("algorithm/algorithm.txt", "r")

   #get data from algorithm.txt
   spaceList = f1.readline()
   f1.close()

   #display the record and write it on a text file
   for a in range((len(table[0].split(",")))-1):
      for b in range(len(table)):
         print((table[b].split(","))[a],end="\t")

         #open the text file and store the records
         f2 = open("record/" + spaceList + ".txt", "a+")
         f2.write(str((table[b].split(","))[a]+"\t"))
```

```
        #open algorithm.txt file and updating
        f1 = open("algorithm/algorithm.txt", "w")
        f1.write(str(int(spaceList) + 1))

    print()
    f2.write("\n")
    f2.close()
print("\n*Your Record Save in ")
print("\n\tRecord: DOC334_CW_2019784 -----> record ----->",spaceList,".txt file")

#import ht.py file in htmlcode folder
from htmlcode.html import html

#calling html def in html.py
html()
```

4) html.py

```
#----------------------VARIABLES------------------------
   #f1      : open algorithm.txt file in algorithm folder
   #spaceList: keep data from file
   #ftitle   : using this to give a name for html code file
   #f2      : open record file in the record folder
   #sf2      : using this to keep data from  f2
   #f3      : create html files in htmlcode/record folder
   #shtml    : using this for keep  html code

def html():
#open files of f1,f2,f3 and acquire data
#write html files

   #open the algorithm.txt file
   f1= open("algorithm/algorithm.txt","r")
   spaceList=f1.readline()

   #search record text file name
   ftitle=str((int(spaceList))-1)
   f2= open("record/"+ftitle+".txt","r")
   sf2=f2.readlines()
   f3 = open("htmlcode/record/"+(str(int(spaceList)-1))+'.html','a')

   #create html tags on htmlcode file
   shtml = '''<html> <body> <h5> DOC334_CW_2019784/htmlcode/record </h5>
      <h3 align = "Center"> Record </h3>
      <table style="width:40%",align:"Center", border=1 , align="Center">'''
```

```
f3.write(shtml)
for b in sf2:
    f3.write("<tr align = center>")
    blank= b.split("\t")

    for a in range((len(blank))-1):
        f3.write("<td>"+blank[a]+"</td>")

    f3.write("</tr>")

shtml = '''
</table>
</body>
</html>'''

f3.write(shtml)
f1.close()
f3.close()
f2.close()

print("\n\n*you can follow the web browser to see the record","\n\n",
      "\tRecord Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record"
      +str(int(spaceList)-1)+".html")
```

Test Cases

Table Of Test Cases

Test Case	Inputs	Expected Output	Actual Output	Remarks
T01	C:\ >perc.py 2x1	*Display the error message to user and ask to follow the condition. *Ask to try back.	------Your Enter Grid Size is Valid------ *Grid 3x3 should be the LOWEST DIMENSIONS for this system *Grid 9x9 should be the HIGHEST DIMENSIONS for this system ------Please Correct the Grid Size and Try Again!------	Test Case Pass
T02	C:\> python perc.py 3x2	*Display the error message to user and ask to follow the condition. *Ask to try back.	------Your Enter Grid Size is Valid------ *Grid 3x3 should be the LOWEST DIMENSIONS for this system *Grid 9x9 should be the HIGHEST DIMENSIONS for this system ------Please Correct the Grid Size and Try Again!------!	Test Case Pass
T03	C:\>python perc.py 3x3	*Display the 3x3 grid with random numbers and condition of "OK" or "NO" statement. * Then the record of the grid display in the text file and html file. *Display a message to user go and check it.	13 39 49 96 85 28 98 74 OK OK NO *Your Record Save in Record: DOC334_CW_2019784 -----> record -----> 77 .txt file *you can follow the web browser to see the record Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record77.html	Test Case Pass

T04	C:\>python perc.py 10x9	*Display the error message to user and ask to follow the condition. *Ask to try back.	------Your Enter Grid Size is Valid------ *Grid 3x3 should be the LOWEST DIMENSIONS for this system *Grid 9x9 should be the HIGHEST DIMENSIONS for this system ------Please Correct the Grid Size and Try Again!------	Test Case Pass
T05	C:\>python perc.py 20x2	*Display the error message to user and ask to follow the condition. *Ask to try back.	------Your Enter Grid Size is Valid------ *Grid 3x3 should be the LOWEST DIMENSIONS for this system *Grid 9x9 should be the HIGHEST DIMENSIONS for this system ------Please Correct the Grid Size and Try Again!------	Test Case Pass
T06	C:\>python perc.py 9X9	*Display the 9x9 grid with random numbers and condition of "OK" or "NO" statement. * Then the record of the grid display in the text file and html file. *Display a message to user go and check it.	46 81 49 83 96 59 44 40 27 84 37 69 69 48 25 93 25 48 69 63 61 42 43 23 41 89 23 37 28 66 16 38 17 88 77 41 93 93 35 67 27 11 35 61 30 10 39 16 55 68 97 83 22 82 56 33 76 28 17 82 98 51 16 10 10 48 27 84 68 22 38 25 15 20 76 41 29 31 96 NO OK OK OK OK OK OK OK NO *Your Record Save in Record: DOC334_CW_2019784 -----> record -----> 91 .txt file *you can follow the web browser to see the record Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record91.html	Test Case Pass
T07	C:\>python perc.py 6X7	*Display the 6x7 grid with random numbers and condition of "OK" or "NO" statement. * Then the record of the grid display in the text file and html file.	45 16 31 67 17 24 98 35 29 12 73 46 63 46 76 15 34 74 32 95 40 12 89 83 96 34 88 64 94 52 51 88 65 57 10 61 60 70 34 33 OK OK NO OK OK OK NO *Your Record Save in	Test Case Pass

		*Display a message to user go and check it.	Record: DOC334_CW_2019784 -----> record -----> 126 .txt file *you can follow the web browser to see the record Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record126.html	
T08	C:\>python perc.py 7X4	*Display the 7x4 grid with random numbers and condition of "OK" or "NO" statement. * Then the record of the grid display in the text file and html file. *Display a message to user go and check it.	97 14 77 12 62 16 90 98 16 79 29 53 15 22 37 49 70 51 13 77 12 32 80 13 76 79 36 NO OK OK OK *Your Record Save in Record: DOC334_CW_2019784 -----> record -----> 130 .txt file *you can follow the web browser to see the record Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record130.html	Test Case Pass
T09	C:\>python perc.py	*Display the 5x5 grid with random numbers and condition of "OK" or "NO" statement. (Default Value) * Then the record of the grid display in the text file and html file. *Display a message to user go and check it.	64 99 37 18 33 22 14 37 20 76 69 18 21 37 47 91 51 59 61 55 97 38 98 NO OK NO OK OK *Your Record Save in Record: DOC334_CW_2019784 -----> record -----> 191 .txt file *you can follow the web browser to see the record Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record191.html	Test Case Pass

T10	C:\>python PERC.PY	*Display the 5x5 grid with random numbers and condition of "OK" or "NO" statement. (Default Value) * Then the record of the grid display in the text file and html file. *Display a message to user go and check it.	46 92 79 22 67 60 75 55 10 59 20 22 46 84 69 53 70 51 44 13 77 92 99 53 OK OK OK OK NO *Your Record Save in Record: DOC334_CW_2019784 -----> record -----> 212 .txt file *you can follow the web browser to see the record Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record212.html	Test Case Pass

Screenshot Of Test Cases

1) Test Case 01 (T01)

i. Output: Command Console

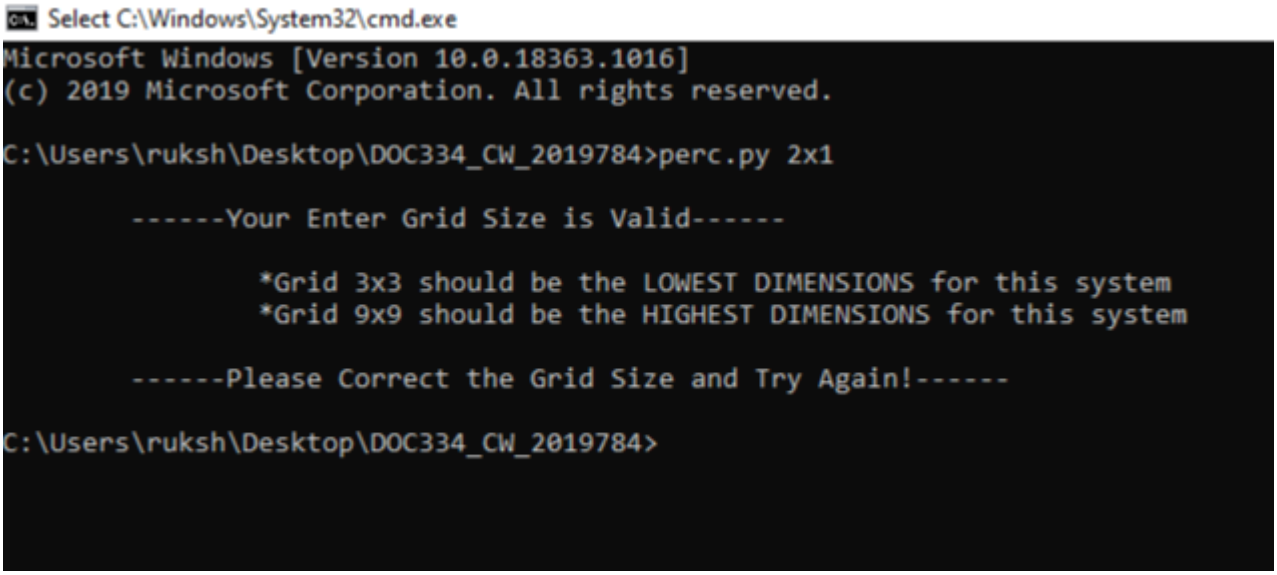


Figure 3: Test Case 01_Command Console

ii. Output: Text File

- User enter value has been invalid so there is no result for that the record.

iii. Output: HTML File

- User enter value has been invalid so there is no result for that the record.

2) Test Case 02 (T02)

i. Output: Command Console

C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.18363.1016]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 3x2

        ------Your Enter Grid Size is Valid------

                *Grid 3x3 should be the LOWEST DIMENSIONS for this system
                *Grid 9x9 should be the HIGHEST DIMENSIONS for this system

        ------Please Correct the Grid Size and Try Again!------

C:\Users\ruksh\Desktop\DOC334_CW_2019784>_
```

Figure 4: Test Case 02_Command Console

ii. Output: Text File

- User enter value has been invalid so there is no result for that the record.

iii. Output: HTML File

- User enter value has been invalid so there is no result for that the record.

3) Test Case 03 (T03)

i. Output: Command Console

C:\Windows\System32\cmd.exe

```
C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 3x3
13      39      49
96      85      28
98      74
OK      OK      NO

*Your Record Save in

        Record: DOC334_CW_2019784 -----> record -----> 77 .txt file


*you can follow the web browser to see the record

        Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record77.html

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

Figure 5: Test Case 03_Command Console

ii. Output: Text File

77 - Notepad

File Edit Format View Help

13	39	49
96	85	28
98	74	
OK	OK	NO

Figure 6: Test Case 03_Text File

iii. Output: HTML File

77.html

File | C:/Users/ruksh/Desktop/DOC334_CW_2019784/htmlcode/record/77.html

DOC334_CW_2019784/htmlcode/record

Record

13	39	49
96	85	28
98	74	
OK	OK	NO

Figure 7: Test Case 03_HTML File

4) Test Case 04 (T04)

i. Output: Command Console

C:\Windows\System32\cmd.exe

```
C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 10x9

        ------Your Enter Grid Size is Valid------

                *Grid 3x3 should be the LOWEST DIMENSIONS for this system
                *Grid 9x9 should be the HIGHEST DIMENSIONS for this system

        ------Please Correct the Grid Size and Try Again!------

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

Figure 8: Test Case 04_Command Console

ii. Output: Text File

- User enter value has been invalid so there is no result for that the record.

iii. Output: HTML File

- User enter value has been invalid so there is no result for that the record.

5) Test Case 05 (T05)

i. Output: Command Console

C:\Windows\System32\cmd.exe

```
C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 20x2

        ------Your Enter Grid Size is Valid------

                *Grid 3x3 should be the LOWEST DIMENSIONS for this system
                *Grid 9x9 should be the HIGHEST DIMENSIONS for this system

        ------Please Correct the Grid Size and Try Again!------

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

Figure 9: Test Case 05_Command Console

ii. Output: Text File

- User enter value has been invalid so there is no result for that the record.

iii. Output: HTML File

- User enter value has been invalid so there is no result for that the record.

6) Test Case 06 (T06)

i. Output: Command Console

```
C:\Windows\System32\cmd.exe

C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 9x9
46      81      49      83      96      59      44      40      27
84      37      69      69      48      25      93      25      48
69      63      61      42      43      23      41      89      23
37      28      66      16      38      17      88      77      41
93      93      35      67      27      11      35      61
        30      10      39      16      55      68      97      83
22      82      56      33      76      28      17      82      98
51      16      10      10      48      27      84      68      22
38      25      15      20      76      41      29      31      96
NO      OK      OK      OK      OK      OK      OK      OK      NO

*Your Record Save in

        Record: DOC334_CW_2019784 -----> record -----> 91 .txt file


*you can follow the web browser to see the record

        Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record91.html

C:\Users\ruksh\Desktop\DOC334_CW_2019784>_
```

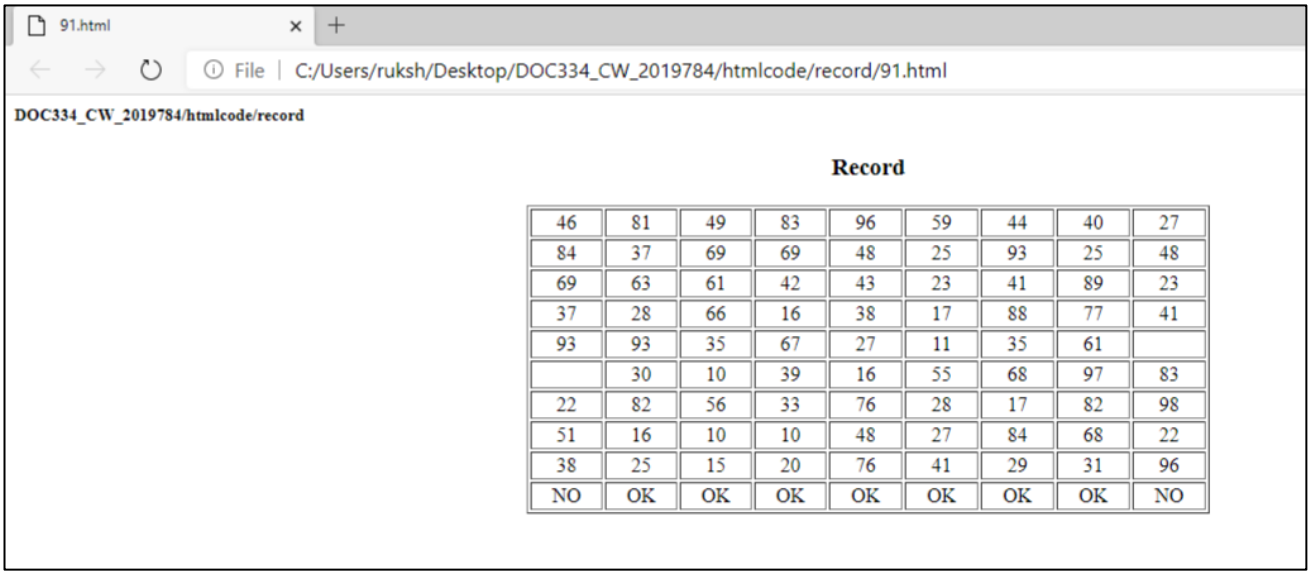
Figure 10: Test Case 06_Command Console

ii. Output: Text File

```
91 - Notepad
File  Edit  Format  View  Help
46      81      49      83      96      59      44      40      27
84      37      69      69      48      25      93      25      48
69      63      61      42      43      23      41      89      23
37      28      66      16      38      17      88      77      41
93      93      35      67      27      11      35      61
        30      10      39      16      55      68      97      83
22      82      56      33      76      28      17      82      98
51      16      10      10      48      27      84      68      22
38      25      15      20      76      41      29      31      96
NO      OK      OK      OK      OK      OK      OK      OK      NO
```

Figure 11: Test Case 06_Text File

iii. Output: HTML File



46	81	49	83	96	59	44	40	27
84	37	69	69	48	25	93	25	48
69	63	61	42	43	23	41	89	23
37	28	66	16	38	17	88	77	41
93	93	35	67	27	11	35	61	
	30	10	39	16	55	68	97	83
22	82	56	33	76	28	17	82	98
51	16	10	10	48	27	84	68	22
38	25	15	20	76	41	29	31	96
NO	OK	OK	OK	OK	OK	OK	OK	NO

Figure 12: Test Case 06_HTML File

7) Test Case 07 (T07)

i. Output: Command Console

C:\Windows\System32\cmd.exe

```
C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 6x7
45      16              31      67      17      24
98      35      29      12      73      46
63      46      76      15      34      74      32
95      40      12      89      83      96      34
88      64      94      52      51      88      65
57      10      61      60      70      34      33
OK      OK      NO      OK      OK      OK      NO

*Your Record Save in

        Record: DOC334_CW_2019784 -----> record -----> 126 .txt file


*you can follow the web browser to see the record

        Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record126.html

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

Figure 13: Test Case 07_Command Console

ii. Output: Text File

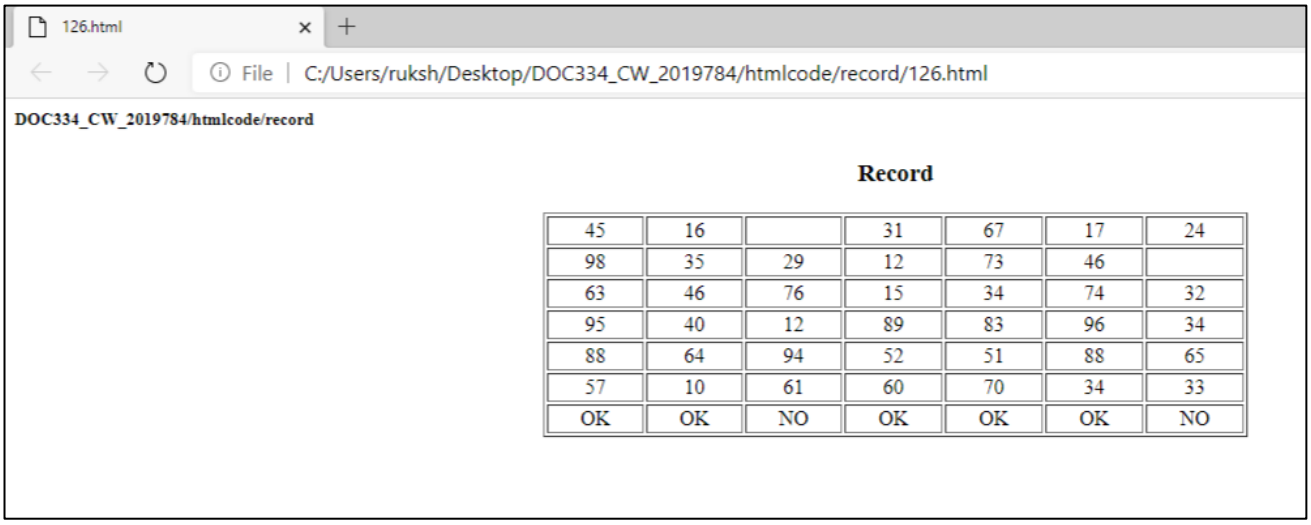
126 - Notepad

File Edit Format View Help

```
45      16              31      67      17      24
98      35      29      12      73      46
63      46      76      15      34      74      32
95      40      12      89      83      96      34
88      64      94      52      51      88      65
57      10      61      60      70      34      33
OK      OK      NO      OK      OK      OK      NO
```

Figure 14: Test Case 07_Text File

iii. Output: HTML File



45	16		31	67	17	24
98	35	29	12	73	46	
63	46	76	15	34	74	32
95	40	12	89	83	96	34
88	64	94	52	51	88	65
57	10	61	60	70	34	33
OK	OK	NO	OK	OK	OK	NO

Figure 15: Test Case 07_HTML File

8) Test Case 08 (T08)

i. Output: Command Console

```
Select C:\Windows\System32\cmd.exe
C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py 7x4
97      14      77      12
62      16      90      98
16      79      29      53
15      22      37      49
70      51      13      77
        12      32      80
13      76      79      36
NO      OK      OK      OK

*Your Record Save in

        Record: DOC334_CW_2019784 -----> record -----> 130 .txt file


*you can follow the web browser to see the record

        Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record130.html

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

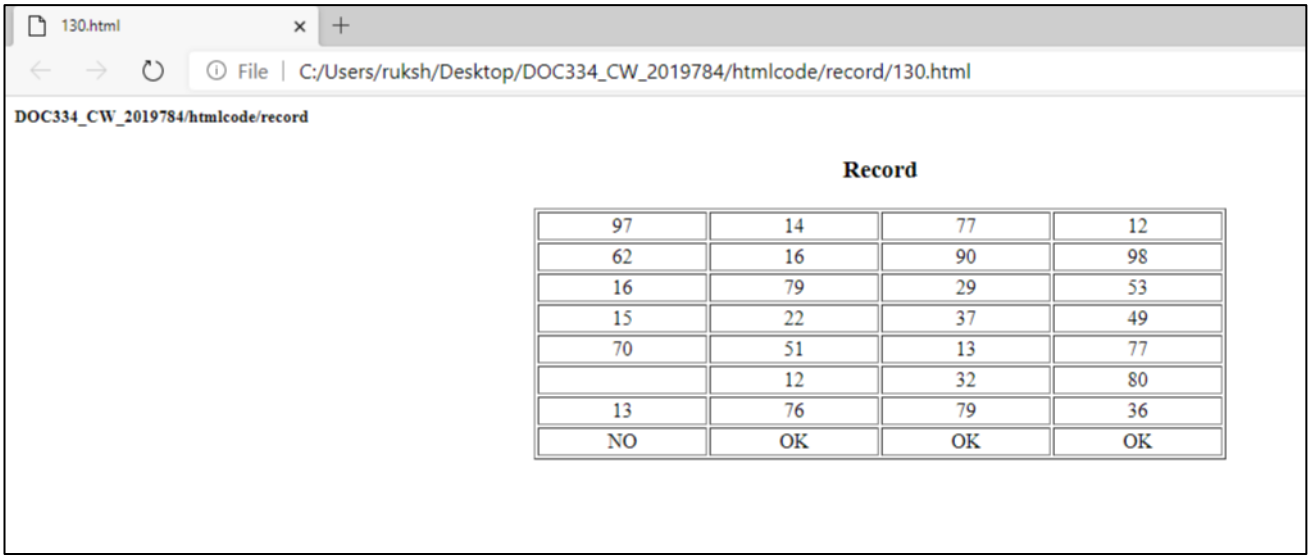
Figure 16: Test Case 08_Command Console

ii. Output: Text File

```
130 - Notepad
File  Edit  Format  View  Help
97      14      77      12
62      16      90      98
16      79      29      53
15      22      37      49
70      51      13      77
        12      32      80
13      76      79      36
NO      OK      OK      OK
```

Figure 17: Test Case 08_Text File

iii. Output: HTML File



97	14	77	12
62	16	90	98
16	79	29	53
15	22	37	49
70	51	13	77
	12	32	80
13	76	79	36
NO	OK	OK	OK

Figure 18: Test Case 08_HTML File

9) Test Case 09 (T09)

i. Output: Command Console

```
C:\Windows\System32\cmd.exe
C:\Users\ruksh\Desktop\DOC334_CW_2019784>perc.py
        64      99      37      18
33      22      14      37      20
76      69              18      21
37      47      91      51      59
61      55      97      38      98
NO      OK      NO      OK      OK

*Your Record Save in

        Record: DOC334_CW_2019784 -----> record -----> 191 .txt file


*you can follow the web browser to see the record

        Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record191.html

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

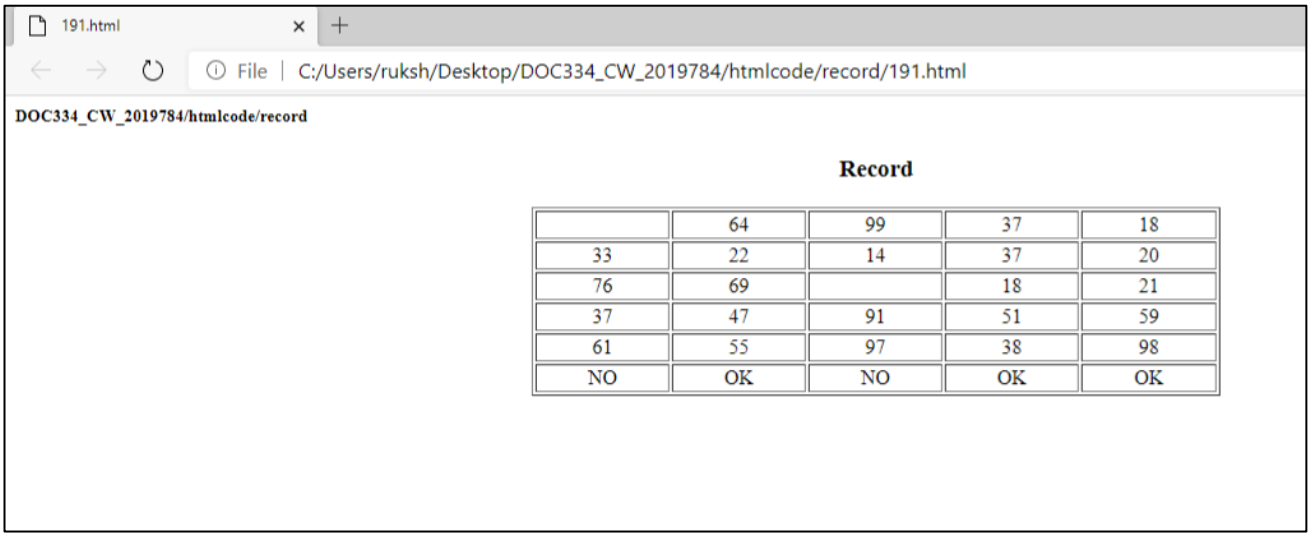
Figure 19: Test Case 09_Console Command

ii. Output: Text File

```
191 - Notepad
File  Edit  Format  View  Help
        64      99      37      18
33      22      14      37      20
76      69              18      21
37      47      91      51      59
61      55      97      38      98
NO      OK      NO      OK      OK
```

Figure 20: Test Case 09_Text File

iii. Output: HTML File



	64	99	37	18
33	22	14	37	20
76	69		18	21
37	47	91	51	59
61	55	97	38	98
NO	OK	NO	OK	OK

Figure 21: Test Case 09_HTML File

10) Test Case 10 (T10)

i. Output: Command Console

```
C:\Windows\System32\cmd.exe
C:\Users\ruksh\Desktop\DOC334_CW_2019784>PERC.PY
46      92      79      22      67
60      75      55      10      59
20      22      46      84      69
53      70      51      44
13      77      92      99      53
OK      OK      OK      OK      NO

*Your Record Save in

        Record: DOC334_CW_2019784 -----> record -----> 212 .txt file


*you can follow the web browser to see the record

        Record Path Direction: DOC334_CW_2019784 -----> htmlcode -----> record212.html

C:\Users\ruksh\Desktop\DOC334_CW_2019784>
```

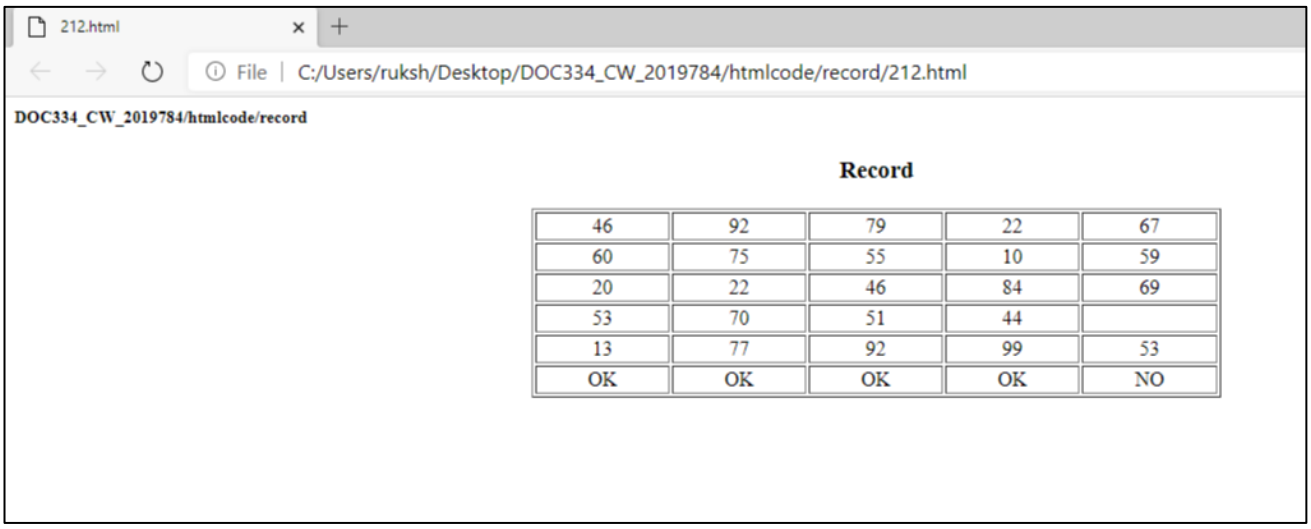
Figure 22: Test Case 10_Console Command

ii. Output: Text File

```
212 - Notepad
File  Edit  Format  View  Help
46      92      79      22      67
60      75      55      10      59
20      22      46      84      69
53      70      51      44
13      77      92      99      53
OK      OK      OK      OK      NO
```

Figure 23: Test Case 10_Text File

iii. Output: HTML File



46	92	79	22	67
60	75	55	10	59
20	22	46	84	69
53	70	51	44	
13	77	92	99	53
OK	OK	OK	OK	NO

Figure 24: Test Case 10_HTML File

End of Coursework