





| Experiment No.5 |
| --- |
| Exploring Files and Directories: Python program data to existing file and then display the entire file |
| Date of performance: 14/02/2024 |
| Date of submission: 21/02/2024 |

















**Code:**

f=open("hi1.txt","w")

str=input("Enter a string: ")

f.write(str)

f.close()

f=open("hi1.txt","r")

str1=f.read()

print(str1)

f.close()

f=open("hi2.txt","w")

print("Enter characters till the symbol @ is pressed")

while str!='@':

str=input()

if(str !='@'):

f.write(str+'\n')

else:

f.close()

f=open("hi1.txt","a+")

print("Enter characters till the symbol @ is pressed")

while str!='@':

str=input()

if(str !='@'):

f.write(str+'\n')

else:

f.seek(0,0)

f=open("hi1.txt","r")

str2=f.read()

print(str2)

f.close()

**Output:**

Hello!This is SarwadeepPython is easy

byee

helloo

python is easy

Hello!This is Sarwadeep.

Python is interesting.

Thank you!

**Conclusion:**

Python provides efficient tools for file and directory operations, including opening, reading, writing, and closing files. Context managers ensure safe and reliable file handling by automatically closing files when they're no longer needed. These features contribute to clean and robust code for working with files in Python.