Task 1

A screenshot of a computer

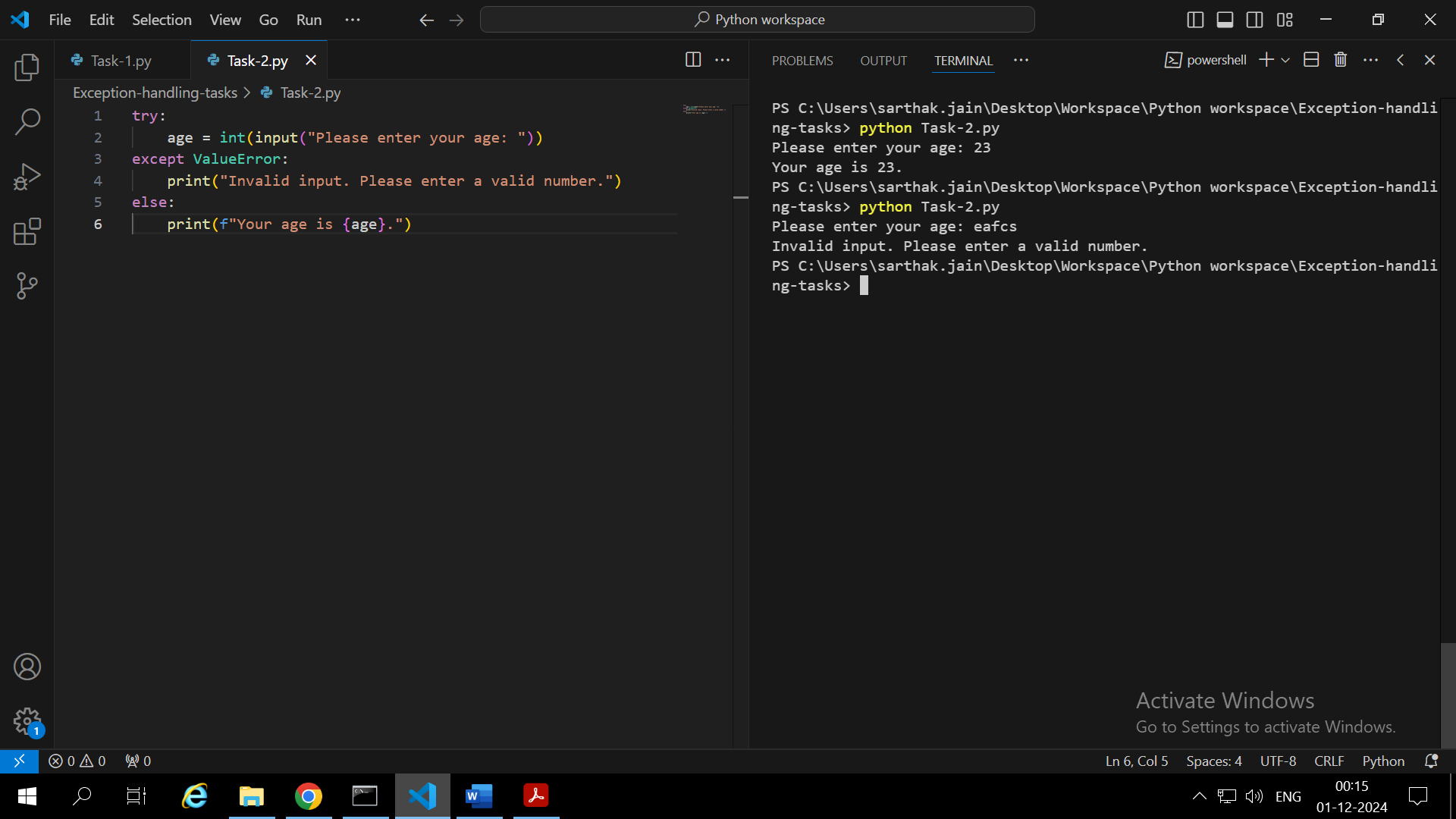
Description automatically generated

ZeroDivisionError exception is raised when trying to divide a number by 0.

When an exception is raised the program prints the message defined in the except block and further execution continues normally. Program does not terminate when an exception is raised.

SyntaxError occurs as python expects except block after try block.

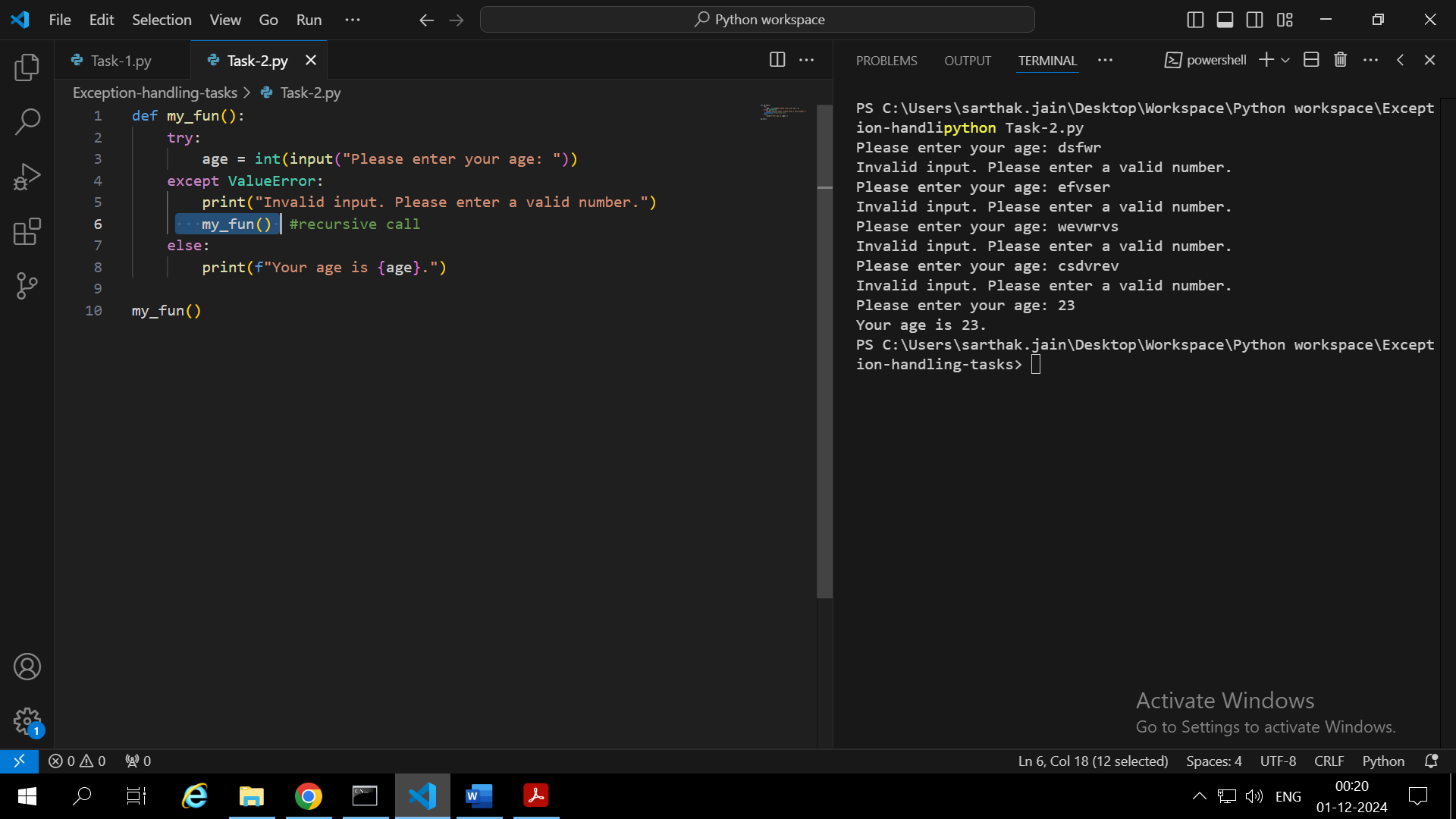
Task 2



int() is used to convert the input number to integer. Since the input() takes input in form of string. So we need to convert it to int.

int(“23” ) 🡪 23

If the python is not able to convert the input to integer. That is someone entered alphabets which can’t be converted to int. So python raises ValueError.



We can use recursion. Wrap the try except block in a function and call function recursively in except block. Thus the user will be prompted to enter the age again and again until he enters a valid age.

Task-3

A screenshot of a computer

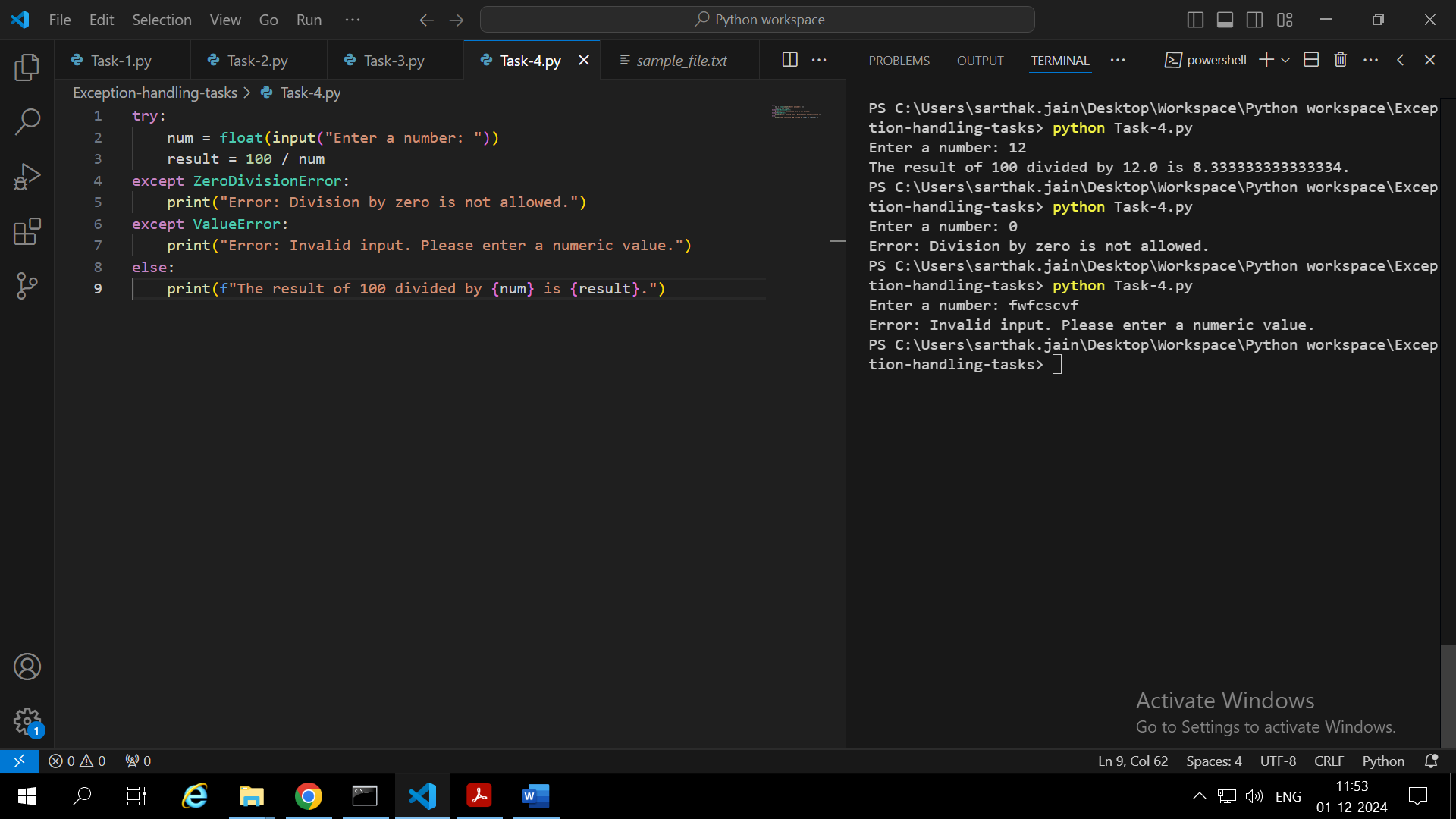
Description automatically generated

If the file is not found in the same directory then “FileNotFoundError” is raised.

“with” is an automatic resource management used in file handling and database connection. It ensures closing resources right after processing them

IOError is another type of error during file handling

Task-4



The purpose of using multiple exception blocks is to handle different types of errors which may arise. Like if we divide by zero then it will be handled by zerodivisionerror block. But if someone enters characters in place of number then we need ValueError block to handle it otherwise program will terminate with error.

A screenshot of a computer

Description automatically generated

OverflowError: when someone enters a very large number in input which can’t be converted to float.

Task-5

A screenshot of a computer

Description automatically generated

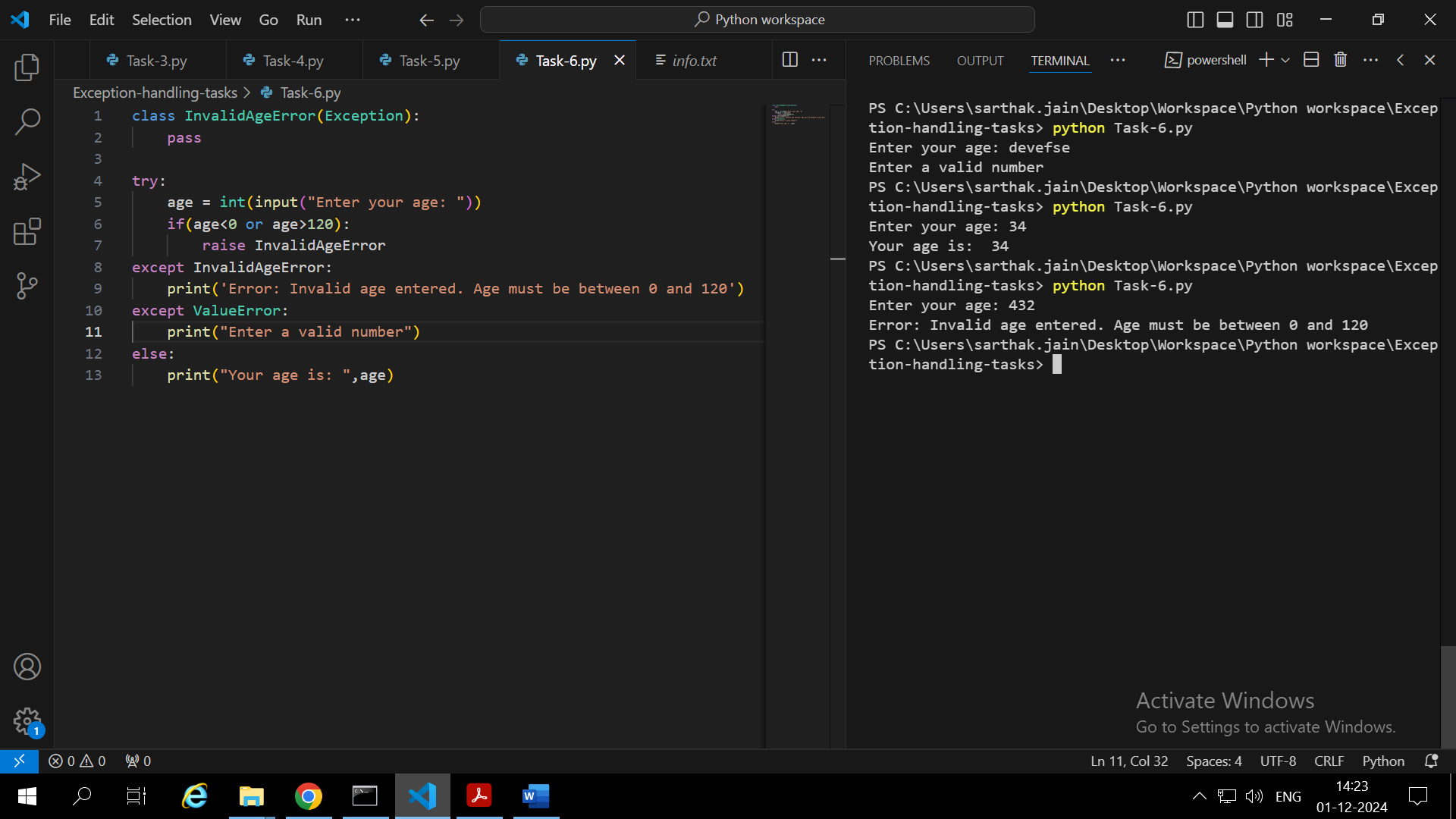
If the finally block is removed the file will automatically get closed when the program will terminate. But if we try to access the file in that program only it will not allow since the file will remain open until the program terminates because we have not manually closed the file using file.close().

Finally block guarantees the file will get closed irrespective of the exception occurred or not because finally() executed whether an exception occurred or not.

**Keeping files open leaves you vulnerable to losing data**. Uf your program—or computer—crashes, the open files can get corrupted.

**Finally** block is used to handle some important tasks after execution of try except. Such as deallocating system resources.

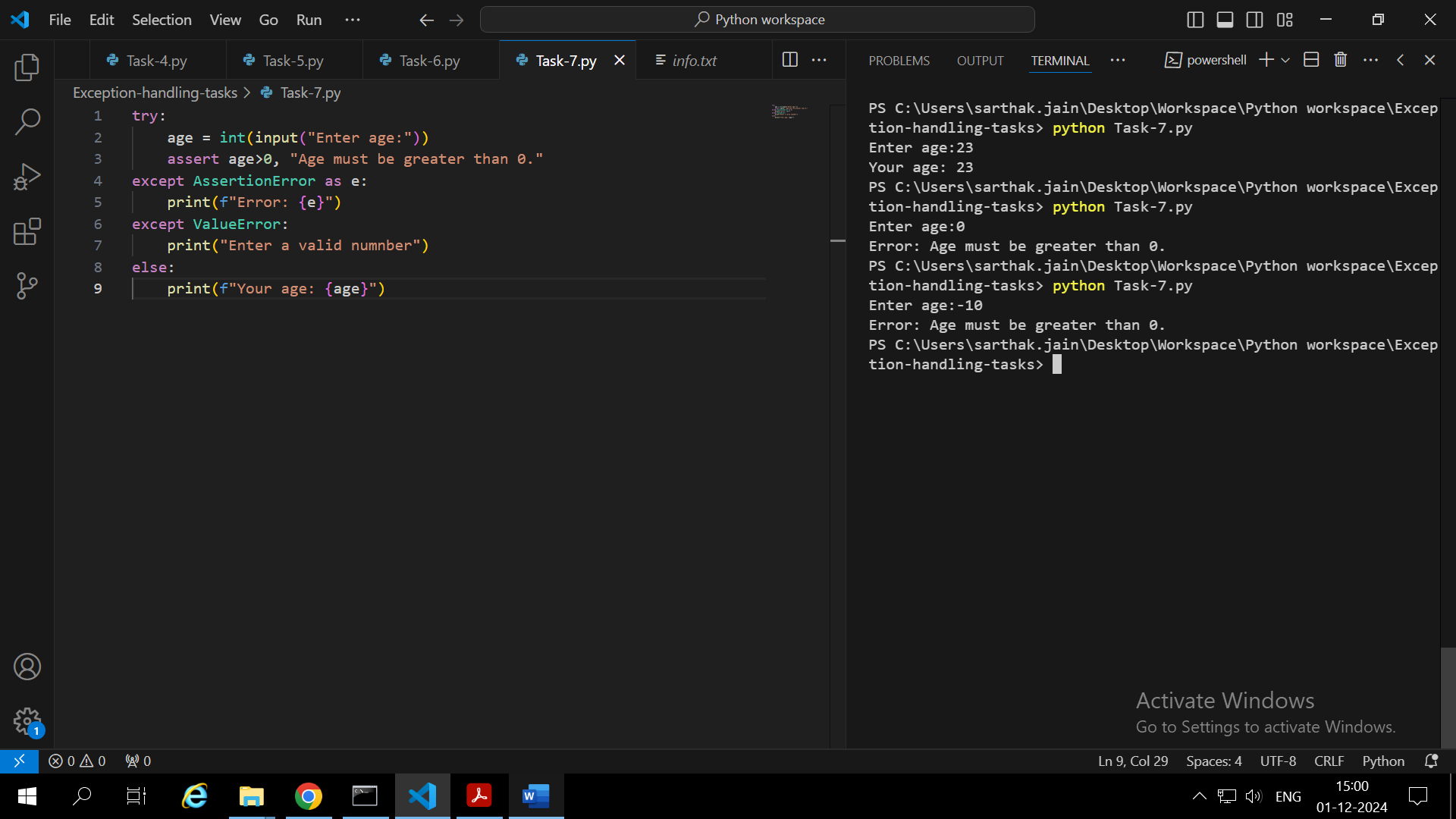
Task-6:



Age <0 or Age > 120 is an error according to us. But according to python age is a number and a number can be <0or >120. So, python will not raise an error in this condition. That’s why we need to define a custom exception to handle this case.

Custom exception allows us to create exceptions wherever necessary and which are not internally defined in python.

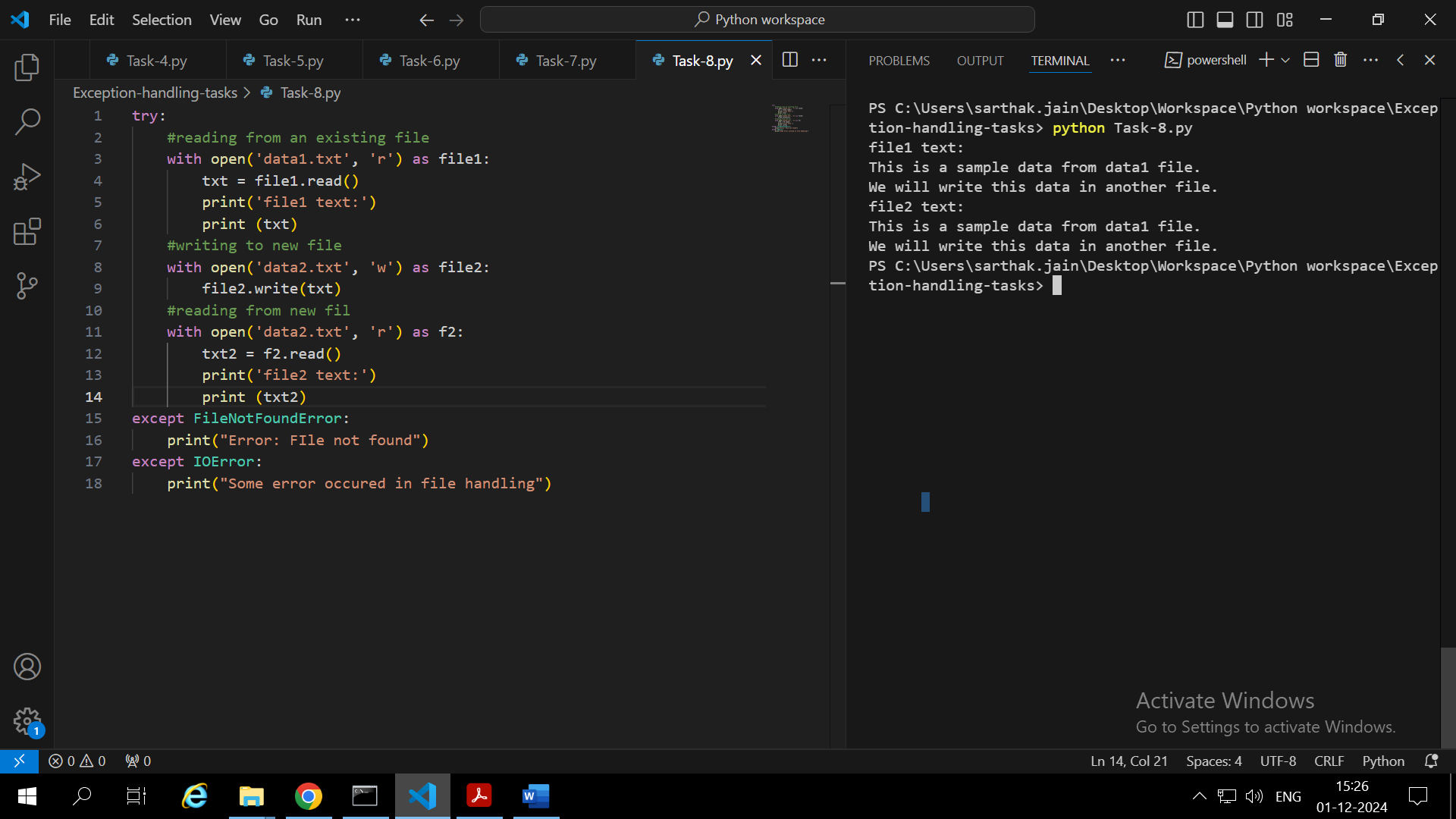
Task-7:



**Assert** is used to validate a condition. If the condition is true then no problem. **Else** **block** will run. Otherwise **except** block will run.

If the non numeric value is entered it will be handled by the ValueError except block.

Task-8:



If the source file does not exist then **FileNotFoundError** will occur which will be handled in except bock.

Using the with statement we don’t need to close the file. With statement automatically closes the file after operation.