**Lab 12 - Files and exceptions**

**1.** Given:

number\_str = input("Input a floating-point number: ")   
while True:   
 # Line 1   
print("Number is",number\_float)

Write a try/except block in # Line 1 that will keep prompting until a correctly formatted floating-point is entered

**2.** Write a function named safe\_input(prompt,type) that works like the Python

input function, except that it only accepts the specified type of input. The function

takes two arguments:

* prompt: str
* type: int, float, str

The function will keep prompting for input until the correct input of the specified type is entered. The function returns the input. If the input was specified to be a number (float or int), the value returned will be of the correct type; that is, the function will perform the conversion. The default for a prompt is the empty string. The default for the type is string.

**3.** Write a function named prompt\_open that prompts for a file name and repeatedly attempts to read the specified file until a correctly specified file has been entered. The function takes one mode argument, 'r' or 'w', and returns the file handle that open returns.

**4.** Write a program that prompts for three numbers. Divide the first number by the

second number and add that result to the third number. Using exceptions check for

the following errors: ValueError, and ZeroDivisionError.

**5.** Given:

*# reverse each line of the input file in the output file*

file\_str = input(**"Open what file:"**)

find\_line\_str = input(**"Which line (integer):"**)

**try**:

input\_file = open(file\_str) *# potential user error*

find\_line\_int = int(find\_line\_str) *# potential user error*

line\_count\_int = 1

**for** line\_str **in** input\_file:

**if** line\_count\_int == find\_line\_int:

print(**"Line {} of file {} is {}"**.format(find\_line\_int, file\_str, line\_str))

**break**

line\_count\_int += 1

**else**:

*# get here if line sought doesn't exist*

print(**"Line {} of file {} not found"**.format(find\_line\_int, file\_str))

input\_file.close()

**except** IOError:

print(**"The file"**, file\_str, **"doesn't exist."**)

**except** ValueError:

print(**"Line"**, find\_line\_str, **"isn't a legal line number."**)

print(**"End of the program"**)

(a) When IOError occurred the

user had to enter a line number before the error occurred. Rewrite the code so that if a

bad file name is entered, the error will be handled before a line number is requested.

(b) Rewrite the code so that if IOError occurs the program keeps asking for input until the user gets it right

(c) Rewrite the code so that if error ValueError occurs the program keeps asking for input until the user gets it right.