**Lab 6 - Files and exceptions**

**Questions:**

**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

**number\_str = input("Input a floating-point number: ")**

**while True:**

**try:**

**number\_float = float(number\_str)**

**break**

**except ValueError:**

**number\_str = input("Invalid input. Please enter a valid floating point number: ")**

**print("Number is",number\_float)**

**2.** Write a function named safe\_input(prompt,type\_input) 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.

**def safe\_input(prompt="", type\_input= str):**

**while True:**

**user\_input = input(prompt)**

**try:**

**if type\_input == int:**

**return int(user\_input)**

**elif type\_input == float:**

**return float(user\_input)**

**elif type\_input == str:**

**return user\_input**

**else:**

**raise ValueError("Unsupported type\_input: ", type\_input)**

**except ValueError:**

**print("Invalid input. Please enter a valid input.")**

**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.

**def prompt\_open(mode):**

**while True:**

**try:**

**file\_name = input(f"Enter the file name to open in {mode} mode: ")**

**return open(file\_name, mode)**

**except FileNotFoundError:**

**print(f"File not found. Please enter a valid file name.")**

**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.

**def calculation():**

**while True:**

**try:**

**num1 = int(input("Enter the first number: "))**

**num2 = int(input("Enter the second number: "))**

**num3 = int(input("Enter the third number: "))**

**result = (num1 / num2) + num3**

**print("The result is:", result)**

**break  # Exit loop if no exceptions are raised**

**except ValueError:**

**print("Please enter a valid integer value.")**

**except ZeroDivisionError:**

**print("Please enter a non-zero number, numbers cannot be divided by zero")**

**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.

**while True:**

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

**try:**

**input\_file = open(file\_str)  # potential IOError (file not found) -- remove this from first line of next try block**

**break  # exit loop if file opens successfully**

**except IOError:**

**print(f"The file '{file\_str}' doesn't exist. Please try again.")**

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

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

**Within part a) ^^**

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

**# add before line\_count\_int = 1**

**while True:**

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

**try:**

**find\_line\_int = int(find\_line\_str)  # Try converting to int**

**break  # Exit the loop when valid integer is entered**

**except ValueError:**

**print(f"'{find\_line\_str}' is not a valid integer. Please enter a valid line number.")**