LAB ACTIVITY 5(ii):

Python Modules and Exception Handling

**Learning Outcomes:**

By the end of this laboratory session, you should be able to:

1. Build Python simple program using Python package
2. Manipulate the implementation of Exception handling in Python coding

**Hardware/Software:** Computer, Phyton 3.5 or above.

**Activity 5J**

Activity Outcome**:** Creating a package.

Procedure:

**Step 1: Create a new folder** named MyApp.

**Step 2:** Inside MyApp, **create a subfolder** with the name 'mypackage'.

**Step 3: Create an empty \_init\_.py** file in the mypackage folder.

**Step 4:** Open Code editor and type the code based on the following code :

def SayHello(name):

print("Hello ", name)

**Step 5:** Save this code in a file named **greet.py**.

**Step 6:** Open a new python file and type the code based on the following code :

def sum(x,y):

return x+y

def average(x,y):

return (x+y)/2

def power(x,y):

return x\*\*y

**Step 7:** Save this code in a file named **functions.py**.

**Step 8:** Open a new python file and type the code based on the following code :

**import mypackage.functions**

print(**mypackage.functions.sum**(3,2))

print(**mypackage.functions.average**(3,2))

print(**mypackage.functions.power**(3,2))

**Step 9:** Save, compile and run the program. Save the program as Act5j1.py. Display the output in the area below.

**Output:**



**Step 10:** Change the way of importing package by open a new python file and type the code based on the following code:

**from mypackage import functions**

print(**functions.sum(10,20))**

print(**functions.average(10,20))**

print(**functions.power(10,20))**

**Step 11:** Save, compile and run the program. Save the program as Act5j2.py. Display the output in the area below.

**Output:**



**Step 12:** Change the way of importing package by open a new python file and type the code based on the following code:

**from mypackage.greet import SayHello**

nama=input("Masukkan nama: ")

**SayHello(nama)**

**Step 13:** Save, compile and run the program. Save the program as Act5j3.py. Display the output in the area below.

**Output:**



**Activity 5K**

Activity Outcome: Creating a simple exception handling.

Procedure:

**Step 1:** Open Code editor and type the code based on the following code :

try:  
   print(x)  
except:  
   print("An exception occurred")

**Step 2:** Save, compile and run the program. Save the program as Act5k.py. Display the output in the area below.

**Output:**



**Activity 5L**

Activity Outcome: Creating a multiple exception handling.

Procedure:

**Step 1:** Open Code editor and type the code based on the following code :

try:  
   print(x)  
except NameError:  
   print("Variable x is not defined")  
except:  
  print("Something else went wrong")

**Step 2:** Save, compile and run the program. Save the program as Act5l.py. Display the output in the area below.

**Output:**



**Activity 5M**

Activity Outcome: Use a tuple of values to specify multiple exceptions in an except clause

Procedure:

**Step 1:** Open Code editor and type the code based on the following code :

keep\_asking = True

while keep\_asking:

try:

x = int(input("Please enter a number: "))

print("Dividing 50 by", x,"will give you :", 50/x)

except (ZeroDivisionError, ValueError, TypeError):

print("Something has gone wrong..")

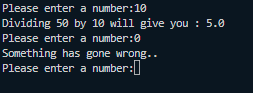
# code to deal with the exception

except:

print("Other than ZeroDivisionError, ValueError, TypeError ")

**Step 2:** Save, compile and run the program. Save the program as Act5m.py. Display the output in the area below.

**Output:**



**Activity 5N**

Activity Outcome : Creating a simple exception handling (‘else’ keyword).

Procedures:

**Step 1:** Open code editor and type the following code:

keep\_asking = True

while keep\_asking:

try:

x = int(input("Please enter a number: "))

except ValueError:

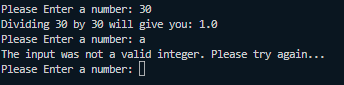
print("The input was not a valid integer. Please try again...")

else:

print("Dividing 30 by", x,"will give you :", 30/x)

**Step 2:**  Save, compile and run the program. Save the program as Act5n.py. Display the output in the area below.

**Output:**



**Activity 5O**

Activity Outcome : Creating a simple exception handling (’finally’ keyword)

Procedures:

**Step 1:** Open code editor and type the following code:

‘’’ The following code checks for two exceptions, TypeError and ValueError. The else block is used to print the factorial.

‘’’

import math

number\_list = [10,-5,1.2,'apple']

for number in number\_list:

try:

number\_factorial = math.factorial(number)

except TypeError:

print("Factorial is not supported for given input type.")

except ValueError:

print("Factorial only accepts positive integer values.", number," is not a positive integer.")

else:

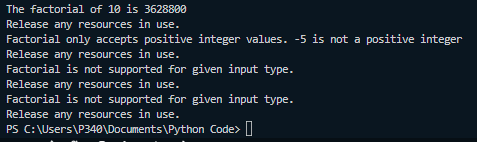
print("The factorial of",number,"is", number\_factorial)

finally:

print("Release any resources in use.")

**Step 2:**  Save, compile and run the program. Save the program as Act5o.py. Display the output in the area below.

**Output:**



**Activity 5P**

Activity Outcome : Creating a simple exception handling (’finally’ keyword)

Procedures:

**Step 1:** Open code editor and type the following code:

#This can be useful to close objects and clean up resources

try:  
  f = open("file.txt")

  f.write("Politeknik METrO Tasek Gelugor")

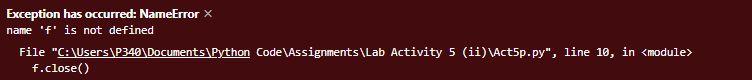
except:  
   print("Something went wrong when writing to the file")

finally:  
  f.close()

**Step 2:**  Save, compile and run the program. Save the program as Act5p.py. Display the output in the area below.

**Output:**





**Step 3:**  Modify the code the statement

f = open("file.txt")

to

f = open("file.txt",**’a’**)

**Step 4:** Save, compile and run the program. Display the output in the area below.

**Output:**





**Activity 5Q**

Activity Outcome : Creating a simple exception handling (**Raise an exception**)

Procedures:

**Step 1:** Open code editor and type the following code:

x = -1  
  
if x < 0:  
  raise Exception("Sorry, no numbers below zero")

**Step 2:**  Save, compile and run the program. Save the program as Act5q.py. Display the output in the area below.

**Output:**

