

GENESIS – Advance Python

./

GENESIS - Learning Outcome & Mini-project Summary Report

# Document History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel.**  **No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **Approved By** | **Remarks/Revision Details** |
| 1.0 | 11/12/2020 | Trupthi.B |  |  |  |
| 2.0 | 12/12/2020 | Trupthi.B |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Contents**

* 1. *ACTIVITY 1 & 2 bank management system CODE ………………………………………….5*

*2.0 ACTIVITY 3 REGULAR EXPRESSION IMPLEMENTATION ………………………………………………10*

**Table of Figures**

[Fig 1.1 Screenshot of all errors being cleared in PEP8 9](#_bookmark11)

Fig 1.2 Screenshot of output of regular expression ………………………………………………………………………10

Activity 1 and Activity 2

Bank management System

# Bank management System

import pickle

import os

class Customer:

def \_\_init\_\_(self,A):

self.name=input("Enter Name: ")

self.type=input("Type of Account s/c?: ")

self.amount=int(input("Enter Amount: "))

while True:

if self.type=='s':

if self.amount<5000:

print("Min 5000 required")

self.amount=int(input("Please Enter amount again: "))

else:

break

if self.type=='c':

if self.amount<2000:

print("Min 2000 required")

self.amount=int(input("Please Enter amount again: "))

else:

break

self.accountNo=A

print("Your Account No. is:",self.accountNo)

def Display(self):

print("{:<15} {:<15} {:<15} {:<15}".format(self.accountNo,self.name,self.type,self.amount))

def createAccount():

try:

file=open('bank.bin','rb')

while True:

t=pickle.load(file)

A=t.accountNo

except FileNotFoundError as e:

A=121000

except EOFError as e:

A=A+1

file.close()

file=open('bank.bin','ab')

s=Customer(A)

pickle.dump(s,file)

file.close()

option()

def ViewAllAccount():

try:

file=open('bank.bin','rb')

print("{:<15} {:<15} {:<15} {:<15}".format('Account No.','Name','Type','Amount'))

while True:

t=pickle.load(file)

t.Display()

except FileNotFoundError as e:

print("\nThere Are No Record")

except EOFError as e:

file.close()

option()

def Deposit():

file=open('bank.bin','rb')

file1=open('tmp.bin','wb')

x=int(input("Enter Bank Account: "))

try:

while True:

t=pickle.load(file)

if t.accountNo==x:

cr=int(input("Enter Deposit Amount: "))

t.amount=t.amount+cr

pickle.dump(t,file1)

except:

pass

finally:

file.close()

file1.close()

os.remove('bank.bin')

os.rename('tmp.bin','bank.bin')

option()

def Withdraw():

file=open('bank.bin','rb')

file1=open('tmp.bin','wb')

x=int(input("Enter Bank Account: "))

try:

while True:

t=pickle.load(file)

if t.accountNo==x:

dr=int(input("Enter Withdraw Amount: "))

if t.type=='s':

while True:

if t.amount-dr<5000:

print("Saving Account Balance can't be below 5000")

dr=int(input("Enter Withdraw Amount: "))

else:

t.amount=t.amount-dr

break

if t.type=='c':

while True:

if t.amount-dr<2000:

print("Currunt Account Balance can't be below 2000")

dr=int(input("Enter Withdraw Amount: "))

else:

t.amount=t.amount-dr

break

pickle.dump(t,file1)

except:

pass

finally:

file.close()

file1.close()

os.remove('bank.bin')

os.rename('tmp.bin','bank.bin')

option()

def Update():

file=open('bank.bin','rb')

file1=open('tmp.bin','wb')

x=int(input("Enter Bank Account: "))

try:

while True:

t=pickle.load(file)

if t.accountNo==x:

name=input("Update Name: ")

t.name=name

pickle.dump(t,file1)

except:

pass

finally:

file.close()

file1.close()

os.remove('bank.bin')

os.rename('tmp.bin','bank.bin')

option()

def Search():

file=open('bank.bin','rb')

x=int(input("Enter Bank Account: "))

try:

while True:

t=pickle.load(file)

if t.accountNo==x:

print("\nName:",t.name)

print("Account Type:",t.type)

print("Amount:",t.amount)

print("Account No.:",t.accountNo)

break

except:

pass

finally:

file.close()

option()

def Exit():

return 0

def option():

print()

print("The services we provide\n1. Create Account\n2. View all Account\n3. Deposit\n4. Withdraw\n5. Update\n6. Search\n7. Exit")

n=int(input())

if n==1:

createAccount()

elif n==2:

ViewAllAccount()

elif n==3:

Deposit()

elif n==4:

Withdraw()

elif n==5:

Update()

elif n==6:

Search()

elif n==7:

Exit()

else:

print("You have selected an Invalid option")

print("Please try again with valid one")

option()

class New:

Exit()

pass

s="WELCOME TO THE BANK"

print(s)

option()

Graphical user interface, text, application

Description automatically generated

Fig 1.1: Screenshot of all errors being cleared in PEP8

Activity 3

Regular expression implementation

import re

result = re.match("[a-z]+@[a-z]+.[a-z]{3}", 'trupthibhyregowda@gmail.com')

print(result)

result = re.search("[a-z]+", 'trupthibhyregowda@gmail.com')

print(result)

result = re.findall("[a-z]+", 'trupthibhyregowda@gmail.com')

print(result)

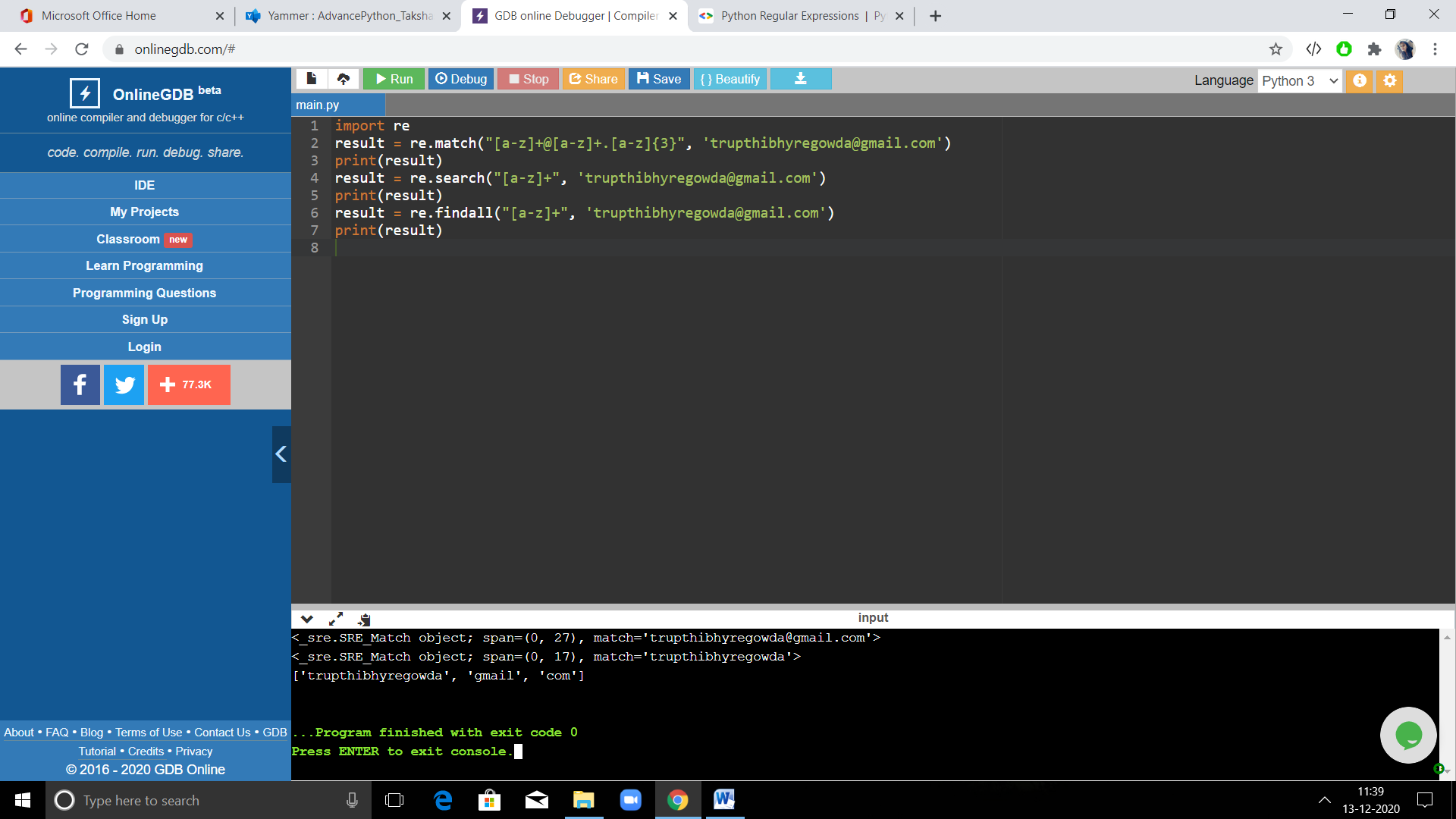


Fig 1.2: Screenshot of output for regular expression