1811ICT/2807ICT/7001ICT Programming Principles Workshop 9

School of Information and Communication Technology

Griffith University

Ghadeer Alkhunaizi

|  |  |
| --- | --- |
| Goals | This workshop focusses on everything in the course up to files. |
| When | Workshops from Friday 20 May to Thursday 26 May |
| Marks | 2 |
| Due | Workshop programming problems by 11:59pm on 27 May |

# Before your workshop class:

* Read all of this document.
* Review the lecture notes sections 1 to 24.
* **Complete the pre-workshop questions posted on the course website**.

# Workshop activities

At any stage, when you are stuck, *ask your tutor*!

## Problem 1

*Problem:* The people of ancient land of Pacific Baza had a simple mathematical system that knew only natural numbers and addition. The genius Bazan scholar, Gringo el Possum, built a computer from wood and various animal parts. Archeologists have recovered ancient scrolls with enough scraps of programs to reconstruct the programming language he named, *Adder*.

The Adder language has only a few simple statements:

|  |  |
| --- | --- |
| quit | Exit the REPL or terminate a program. |
| input *var* | Prompt for and allow the user to enter a value for the variable named *var*. |
| print *val* | Print the value *val*. |
| *var* gets *val* | variable *var* is assigned the value *val*. |
| *var* adds *val* | variable *var* has the value *val* added to it. |

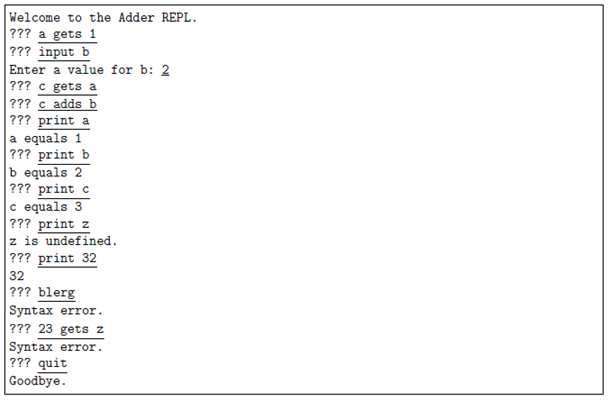
where:

* *var* is always a variable name that contains only letters; and
* *val* can be either:

**–** a variable name that contains only letters; or

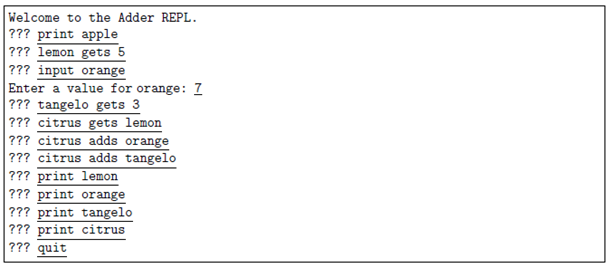
**–** a natural number that contains only digits.

The Adder REPL allows the user to enter commands interactively. For example:



Write the program for the Adder REPL. Hints: Make good use of string methods. Can you divide your program up into smaller pieces by defining functions?

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following scenario:



***Copy your code here***

*import os*

*# Function to get the numerical value from a variable or a direct number*

*def getNum(var\_dict, val):*

*try:*

*return int(val)*

*except ValueError:*

*return var\_dict.get(val, "error")*

*# Function to check if a string is a valid variable name (only contains letters)*

*def isLetter(s):*

*return s.isalpha()*

*# Function to check if a string is a number*

*def isNum(s):*

*return s.isdigit()*

*# Function to handle the REPL commands*

*def handle\_command(command, var\_dict):*

*parts = command.strip().split()*

*if not parts:*

*print("Syntax error.")*

*return*

*cmd = parts[0]*

*if cmd == "quit":*

*print("Goodbye")*

*os.\_exit(0)*

*elif cmd == "input":*

*if len(parts) != 2 or not isLetter(parts[1]):*

*print("Syntax error.")*

*return*

*var\_name = parts[1]*

*try:*

*value = int(input(f"Enter a value for {var\_name}: "))*

*var\_dict[var\_name] = value*

*except ValueError:*

*print("Invalid input. Please enter a natural number.")*

*return*

*elif cmd == "print":*

*if len(parts) != 2:*

*print("Syntax error.")*

*return*

*val = parts[1]*

*num = getNum(var\_dict, val)*

*if num == "error":*

*print(f"{val} is undefined")*

*else:*

*print(num)*

*elif len(parts) == 3 and parts[1] == "gets":*

*var\_name = parts[0]*

*val = parts[2]*

*if not isLetter(var\_name) or not (isLetter(val) or isNum(val)):*

*print("Syntax error.")*

*return*

*num = getNum(var\_dict, val)*

*if num == "error":*

*print(f"{val} is undefined")*

*else:*

*var\_dict[var\_name] = num*

*elif len(parts) == 3 and parts[1] == "adds":*

*var\_name = parts[0]*

*val = parts[2]*

*if not isLetter(var\_name) or not (isLetter(val) or isNum(val)):*

*print("Syntax error.")*

*return*

*if var\_name not in var\_dict:*

*print(f"{var\_name} is undefined")*

*return*

*num = getNum(var\_dict, val)*

*if num == "error":*

*print(f"{val} is undefined")*

*else:*

*var\_dict[var\_name] += num*

*else:*

*print("Syntax error.")*

*# Main function to run the REPL*

*def adder\_repl():*

*var\_dict = {}*

*print("Welcome to the Adder REPL.")*

*while True:*

*try:*

*command = input("> ")*

*handle\_command(command, var\_dict)*

*except (EOFError, KeyboardInterrupt):*

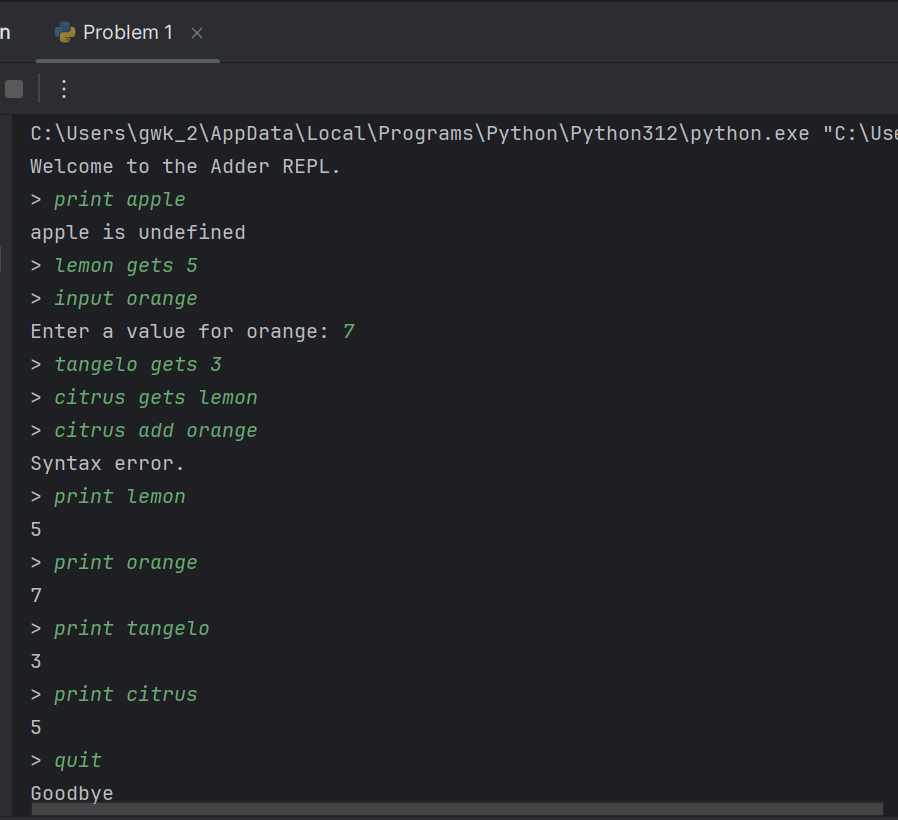
*print("\nGoodbye")*

*break*

*# Run the REPL*

*adder\_repl()*

***Insert your screenshot here***

******

## Problem 2

*Problem:* Write an Adder interpreter, that prompts for and executes an Adder script. For example if the file children.ad contains:

input sons   
input daughters   
children gets sons   
children adds daughters   
print children   
quit

The interpreter would run like this

Script name: children.ad

Enter a value for sons: 3   
Enter a value for daughters: 4   
children equals 7

Hint: This should involve a few small modifications to your REPL from Problem 1.

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following scenario:

* Use the file P2.ad as the source file.

***Copy your code here***

*import os*

*dict={}*

*FileName = input("Script name: ")*

*File = open(FileName, "r")*

*def getNum(dict,para):*

*try:*

*para = int(para)*

*except:*

*try:*

*para = dict[para]*

*except:*

*para ="error"*

*return para*

*def isLetter(S):*

*for p in S:*

*if "a"<=p<="z" or "A"<=p<="Z":*

*pass*

*else:*

*return 0*

*return 1*

*def isNum(dict,para):*

*try:*

*para = int[para]*

*return 1*

*except:*

*return 0*

*def isDictpara(dict,para):*

*return para in dict*

*def handleSql(sql,dict):*

*sql = sql.strip().split(" ")*

*if sql[0] == "quit":*

*print("Exiting the script.")*

*os.\_exit(0)*

*elif sql[0] == "input":*

*s = int(input(f"Enter a value for {sql[1]}: "))*

*dict[sql[1]] = s*

*elif sql[0] == "print":*

*num = getNum(dict, sql[1])*

*if str(num) == "error":*

*print(f"Variable {sql[1]} is not defined.")*

*else:*

*print(f"{sql[1]} equals {num}")*

*elif sql[1] == "gets":*

*if isLetter(sql[0]) and isNum(sql[2]):*

*dict[sql[0]] = getNum(dict, sql[2])*

*elif sql[1] == "adds":*

*if isDictpara(dict, sql[0]) and isNum(sql[2]):*

*dict[sql[0]] += getNum(dict, sql[2])*

*else:*

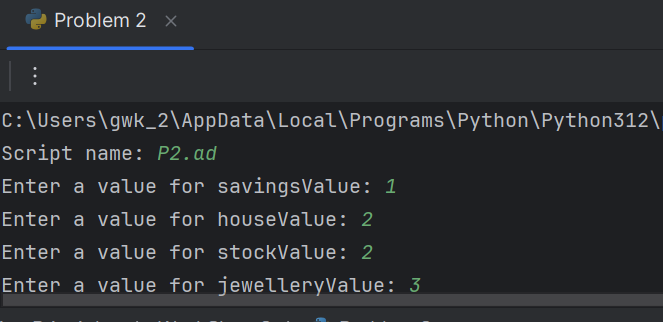
*print("Unknown command")*

*for line in File:*

*handleSql(line, dict)*

*File.close()*

***Insert your screenshot here***

s

## Problem 3

*Problem:* A road map defines locations as map references like B3, where B is the *x*-coordinate value and 3 is the *y*-coordinate.

A

B

C

D

E

F

Y

Z

1

2

3

4

5

6

25

26

…

⫶

B3

E2

Y25

The grid lines are 0.5 km apart.

Write a program that allows the user to enter a trip as a sequence of any number of map references on one line, and reports the total length of the trip, assuming they can travel in straight lines. For example:

Enter trip map references: C2 B5 Y25

Total distance = 16.8 km

For badly formatted map references, your program should exit, reporting the first bad map reference.

Enter trip map references: E6 E4 D7 d43 F5

Bad reference: d43

Hints: You need to *split* the input line into separate references; each reference starts with one character which must be an upper case letter, and the rest must be only digits; and Pythagoras will help. The function exit() can abort the program if you detect an error in the input.

*Answer*: Copy your code in the space given below and insert screenshots of your program output for the following two scenarios:

* A1 B2 C3 D20 S15 W25 Z26
* D2 F23 Ja E23 Z2

***Copy your code here***

*s = input("Enter trip map references")*

*list = s.strip().split()*

*import math*

*for i in range(len(list)):*

*if len(list[i])>= 2 and "A"<=list[i][0]<="Z":*

*try:*

*num = int(list[i][1:])*

*if 1<= num <= 26:*

*pass*

*else:*

*print("Bad reference: %s" % list[i])*

*exit()*

*except:*

*print("Bad reference %s" % list[i])*

*exit()*

*else:*

*print("Bad reference %s" % list[i])*

*exit()*

*total = 0*

*for i in range(1,len(list)):*

*disx= ord(list[i][0])-ord(list[i-1][0])*

*disy= int(list[i][1:])-int(list[i-1][1:])*

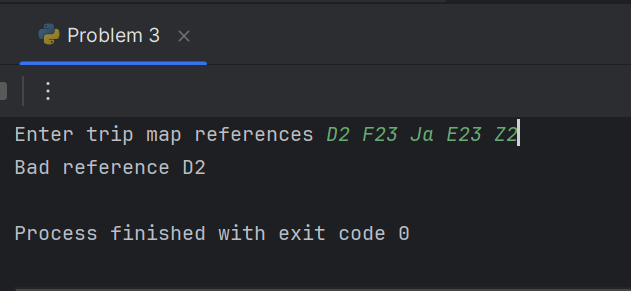
*dis= math.sqrt(disx\*disx+disy\*disy)*

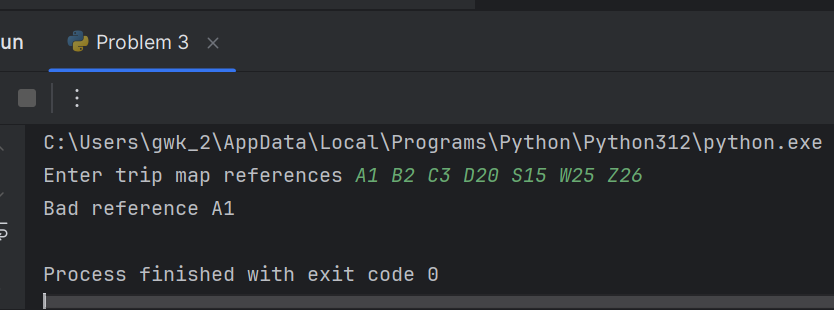
*total += dis*

*total=total\*0.5*

*print("Total distance = %.1f km" % total)*

***Insert your screenshot here***





# Submission and marking

Please submit this document with copied codes and inserted screenshots using the provided submission link in the course website. Students get 2 marks if they complete two or more problems correctly, or 1 mark if they complete one problem correctly, or 0 marks without any attempt.