|  |
| --- |
|  |

|  |
| --- |
| **QA Consulting.** |
| Control Flow |
| Exercise Guide 4 |

# Exercise 4.1 – Calculator

Create:

* New folder and move to: **04ControlFlow\Exercise**
* Create new file: **01Calculator.py**

Code a program that:

* Inputs two numbers
* Outputs menu to select from following operations:
  + Add
  + Subtract
  + Multiply
  + Divide
  + Square
  + Power
  + Calculates and outputs the result

Save and run.

# Exercise 4.2 – Times Table

Create:

* New file: **02TimesTable.py**

Code a program that:

* Calculates and displays the four times table from
  + 4 x 0 = 0 to 4 x 12 = 48
* Modify to calculate and display the five times table from
  + 5 x 0 = 0 to 5 x 12 = 60
* Modify to input an integer between 1 and 12 to calculate and display the appropriate times table
* Modify to output all the times tables for the integers from 1 to 12
  + Note you will need two loops

Save and run.

# Exercise 4.3 – Squares

Create:

* New file: **03Squares.py**

Code a program that:

* Starts at 1 and ends at 100
* Calculates and displays each number and its square
* Until that square is bigger than 2000

Save and run.

# Exercise 4.4 – Factorial

Create:

* New file: **04Factorial.py**

Code a program that:

* Inputs an integer and outputs its factorial
  + Note the factorial of a number is that number multiplied by all the preceding numbers
  + For example, 5! = 5 x 4 x 3 x 2 x 1 = 120

Save and run.

# Exercise 4.5 – Count Vowels

Create:

* New file: **05CountVowels.py**

Code a program that:

* Inputs a word and count how many vowels are in the word

Save and run.

# Exercise 4.6 – Investment

Create:

* New file: **06Investment.py**

Code a program that:

* Calculates how many years it will take an initial investment of £100 to grow to a target value of £1000 if the interest rate is 10%
* Modify to input a variable:
  + Initial investment
  + Target value
  + Interest rate
  + Opens

Save and run.

# Exercise 4.7 – Exam Grade

Create:

* New file: **07ExamGrade.py**

Code a program that:

* Includes a procedure to output the grade for an exam level and mark
* Pass the procedure two parameters
* Level between 1 and 4 inclusive
* Mark between 0 and 100 inclusive
* Grade rules for levels are:
  + 1 and 2: Pass >= 65, Merit >= 75, Distinction >= 85
  + 3: Pass >= 60, Merit >= 70, Distinction >= 80
  + 4: Pass >= 50, Merit >= 60, Distinction >= 70
* If Level or Mark are invalid – Output appropriate message

Save and run.

# Exercise 4.8 – Exam Average

Create:

* New file: **08ExamAverage.py**

Code a program that:

* Includes a procedure to calculate the average of three exam marks
* If the average mark is:
  + >= 65 output a “Pass”
  + < 65 output a “Fail”
* In the main body of the program input the marks for a student for their Maths, English and ICT exams
* Marks should be an integer between 0 and 100
* Use the procedure to calculate their average mark and overall result
* Modify the procedure to return the average mark and print out in the main program

Save and run.

# Exercise 4.9 – Volume

Create:

* New file: **09Volume.py**

Code a program that:

* Includes a procedure to multiple three numbers together and output the result
* In the main body of the program input the lengths of the box: width, depth and height
* Lengths should allow decimal places
* Use the procedure to calculate their volume of the box
* Modify the procedure to a function to return the volume and output the volume in the main program

Save and run.

# Exercise 4.10 – Pythagoras

Create

* New file: **10Pythagoras.py**

Code a program that:

* Calculates the lengths of sides of a triangle using Pythagoras’s Theorem
* See full specification below

Save and run.

### Full Specification

Pythagoras’ Theorem states that the square of the long side (c) of a right-angled triangle is the sum of the squares of the two shorter sides (a and b).

* c\*\*2 = a\*\*2 + b\*\*2

Write a program that will do the following:

* Print a menu:

Pythagoras’ Calculator

* 1 - Find the length of a given b and c
* 2 - Find the length of b given a and c
* 3 – Find the length of c given a and b
* 9 - Exit

Enter an option:

* If ‘1’ is entered, prompt for the length of sides: b and c, calculate the length of side: a and print the answer; Reprint the menu.
* If ‘2’ is entered, prompt for the length of sides: a and c, calculate the length of side: b and print the answer; Reprint the menu.
* If ‘3’ is entered, prompt for the length of sides: a and b, calculate the length of side: c and print the answer; Reprint the menu.
* If ‘9’ is entered, print a goodbye message and exit the program.
* If another value is entered, print an error message and print the menu again.

Note:

* Remember you will need to import the math module (import math) and use the sqrt() function.

# Exercise 4.11 – Time Calculator

Create

* New file: **11TimeCalculator.py**

Code a program that:

* Calculates a range of calculation on times
* See full specification below

Save and run.

### Full Specification

Write a program that will do the following:

* Print a main menu:

Time Calculator - Arithmetic Mode

* 1 - Add 2 times
* 2 - Find the difference between 2 times
* 3 - Convert Time to Days
* 4 - Convert Time to Hours
* 5 - Convert Time to Minutes
* 6 - Convert Minutes to Time
* 7 - Convert Hours to Time
* 8 - Convert Days to Time
* 9 - Exit

Enter an option:

* Complete all the required procedures.
* Times should be stored as strings in the format DD:HH:MM.
* Days, Hours and Minutes should be stored as integers.

# Exercise 4.12 – ATM Basic

Create

* New file: **12ATMBasic.py**

Code a program that:

* Simulates the behaviour of a basic Automatic Teller Machine (ATM) or Cash Machine
* See full specification below

Save and run.

### Full Specification

Northern Frock needs you to write a program for their new ATMs (or Automatic Teller Machines). Assuming the starting balance is £67.14, Write a program that will do the following:

* Print a main menu:

Welcome to Northern Frock

* 1 - Display balance
* 2 - Withdraw funds
* 3 - Deposit funds
* 9 - Return card

Enter an option:

* If ‘1’ is entered, display the current balance and the maximum amount available for withdrawal (must be a multiple of £10), and return to main menu.
* If ‘2’ is entered, print a sub-menu with withdrawal amounts of:

Please select withdrawal amount

* 1 - £10
* 2 - £20
* 3 - £40
* 4 - £60
* 5 - £80
* 6 - £100
* 7 - Other amount
* 8 - Return to main menu

Enter an option:

* + If ‘1 to 6’’ is selected check that the requested withdrawal is allowed, print a message to show that the money has been withdrawn, calculate the new balance and return to main menu.
  + If ‘7’ is selected, then prompt the user for an integer value. Check this number is a multiple of 10 and that the withdrawal is permitted, print a message to show that the money has been withdrawn, calculate the new balance and return to main menu.
  + If ‘8’ is selected return to main menu.
* If ‘3’ is entered, provide another menu that will allow the user to enter an amount to deposit (does not need to be a multiple of £10), return to main menu or return card. If funds are deposited, provide appropriate feedback and update the balance and return to main menu.
* If ‘9’ is entered, print a goodbye message and exit (break).
* If another value is entered, print an error message and print the menu again.

# Exercise 4.13 – Binary Search

Create

* New file: **13BinarySearch.py**

Code a program that:

* Performs a binary search
* See full specification below

Save and run.

### Full Specification

A binary search is the most efficient way to find a value. It involves splitting the available values into two equal halves and testing which half it is in, and then refining this until the correct value is found.

For example, a number must be between 1 and 100, in this case it is 53:

* Midpoint between 1 and 100 is 50 [**int((1+100)/2)**]. Target number is higher.
* Midpoint between 50 and 100 is 75. Target number is lower.
* Midpoint between 50 and 75 is 62. Target number is lower.
* Midpoint between 50 and 62 is 56. Target number is lower.
* Midpoint between 50 and 56 is 53. Target number is found.

Write a program that will do the following:

* Prompt the user for an integer between 1 and 100 (validating and prompting if invalid)
* Use a selection statement to test the value 50 (below, equal, above) and print this guess.
* Repeat the above until the correct number is found.
* Ask the user if they would like to run the program again or quit.

Extensions:

* 1: Count and output the number of steps required to find the number.
* 2: Generate a random number as the target.
* 3: Print all the values that require exactly 7 steps to find them.
* 4: Find the mean average number of steps required to find every number from 1 to 100.
* 5: Calculate the maximum and mean number of steps required if the range of numbers is 200, 1,000, 2,000, 10,000 & 100,000. Are you surprised by the results?

# Exercise 4.14 – Advanced Strings

Create

* New file: **14AdvancedStrings.py**

Code a program that:

* Performs various functions on strings
* See full specification below

Save and run.

### Full Specification

You can use the following functions to do interesting things with strings:

* string = “Strings are sequences of letters”
* x = len(string)
* # Returns the length of the string x = 32
* x = string[0]
* # Returns the 1st character of the string x = S
* x = string[3]
* # Returns the 4th character of the string x = i
* x = string[3:5]
* # Returns the 4th and 5th characters of the string x = in
* x = string[3:6]
* # Returns the 4th - 6th characters of the string x = ing
* x = str.lower(string[0])
* # Returns a lower case string x = s
* x = str.upper(string[3])
* # Returns an upper case string x = I
* x = str.islower(string[3])
* # Returns a bool depending on whether the character is lower case # x = True
* x = str.isupper(string[0])
* # Returns a bool depending on whether the character is lower case # x = True
* x = str.isdigit(string[5])
* # Returns a bool depending on whether the character is a digit
* # x = False
* x = str.isspace(string[7])
* # Returns a bool depending on whether the character is a space
* # x = True

Write a program that will do the following:

* Prompt the user for a string.
* Calculate and print:
  + The number of characters (with spaces).
  + The number of characters (without spaces).
  + The number of upper case characters.
  + The number of lower case characters.
  + An upper-case version of the string.
  + A lower-case version of the string.
  + Whether the string is a palindrome (the same backwards as forwards).
  + Print the string in reverse.

# Exercise 4.15 – Bubble Sort

Create

* New file: **15BubbleSort.py**

Code a program that:

* Performs a bubble sort
* See full specification below

Save and run.

### Full Specification

A bubble sort is a simple (but not particularly efficient) method of putting a list in order.

A bubble sort looks at the first two elements in a list (or array) to see if the first is bigger than the second. If it is, they get swapped round. Either way, the 2nd and 3rd are checked, then the 3rd and 4th, etc...

At the end, the whole thing is done again.

Only when a complete run through happens without any swaps is the list correct.

For example:

* x = [3,9,5,2]
  + Compare x[0] and x[1], 3 < 9 so no swap
  + Compare x[1] and x[2], 9 !< 5 so swap [3,5,9,2]
  + Compare x[2] and x[3], 9 !< 2 so swap [3,5,2,9]
* Run again:
  + Compare x[0] and x[1], 3 < 5 so no swap
  + Compare x[1] and x[2], 5 !< 2 so swap [3,2,5,9]
  + Compare x[2] and x[3], 5 < 9 so no swap
* Run again:
  + Compare x[0] and x[1], 3 !< 2 so swap [2,3,5,9]
  + Compare x[1] and x[2], 3 < 5 so no swap
  + Compare x[2] and x[3], 5 < 9 so no swap
* Run again:
  + Compare x[0] and x[1], 2 < 3 so no swap
  + Compare x[1] and x[2], 3 < 5 so no swap
  + Compare x[2] and x[3], 5 < 9 so no swap
* No swaps, so sorting is complete!

The random module lets you generate random numbers (actually pseudo-random, but that’s another story) using the function random.randint(x,y) where x and y are the lower and upper boundaries.

Write a program that will do the following:

* Generate an array of 6 random integers.
* Print the list, unsorted.
* Perform a bubble sort.
* Print the sorted list.

# Solutions

## Solution 4.1 – Calculator

# Name : 01Calculator

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise calculator

number1 = float(input("Please enter first number : "))

number2 = float(input("Please enter second number: "))

print("Menu:")

print("1 - Add")

print("2 - Subtract")

print("3 - Multiply")

print("4 - Divide")

print("5 - Square")

print("6 - Power")

menu\_option = int(input("Enter Option: "))

if menu\_option == 1:

print(number1,"+",number2,"=",number1 + number2)

elif menu\_option == 2:

print(number1,"-",number2,"=",number1 - number2)

elif menu\_option == 3:

print(number1,"\*",number2,"=",number1 \* number2)

elif menu\_option == 4:

print(number1,"/",number2,"=",number1 / number2)

elif menu\_option == 5:

print(number1,"squared =",number1 \* number1)

print(number2,"squared =",number2 \* number2)

elif menu\_option == 6:

print(number1,"to power of",number2,"=",number1 \*\* number2)

else:

print("Invalid option selected")

# Solution 4.2 – Times Tables

# Name : 02ATimesTable4s

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise of four times table

for i in range(13):

print("4 \*",i," = ",4 \* i)

# Name : 02BTimesTable5s

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise of times table fives

for i in range(13):

print("5 \*",i," = ",5 \* i)

# Name : 02CTimesTableSelect

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise of times table selected from 1 to 12

timestable = int(input("Please enter number between 1 and 12: "))

if timestable >= 1 and timestable <= 12:

for i in range(13):

print(timestable," \* ",i," = ",timestable \* i)

else:

print("Invalid number selected")

# Name : 02DTimesTable1to12

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise times tables for 1 to 12

for i in range(1,13):

for j in range(1,13):

print(i," \* ",j," = ",i \* j)

## Solution 4.3 – Square

# Name : 03Square

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise to calculate squares up to 2000

for i in range(1,101):

square = i \*\* 2

if square < 2000:

print(i,"squared = ",square)

else:

break

## Solution 4.4 – Factorial

# Name : 04Factorial

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise to calculate factorial

integer = int(input("Please enter integer: "))

factorial = 1

for i in range(integer,0,-1):

factorial = factorial \* i

print("factorial of",integer,"=",factorial)

## Solution 4.5 – Count Vowels

# Name : 05CountVowels

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise to count vowels in word

word = str(input("Please enter a word: "))

count\_vowel = 0

for char in word:

if char == "a" or char == "e" or char == "i" or char == "o" or char == "u":

count\_vowel = count\_vowel + 1

print("Number of vowels in",word,"=",count\_vowel)

vowels = ["a","e","i","o","u"]

count\_vowel = 0

for char in word:

if char in vowels:

count\_vowel = count\_vowel + 1

print("Number of vowels in",word,"=",count\_vowel)

## Solution 4.6 – Investment

# Name : 06Investment

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise to calculate investment

investment = 100.0

year = 0

while investment < 1000.0:

year = year + 1

investment = investment \* 1.1

print("year",year,"investment",investment)

## Solution 4.7 – Exam Grade

# Name : 07ExamGrade

# Author : John Merchant

# Date : 12 Jul 2016

# Purpose : Exercise to determine exam grade

def exam\_grade(level,mark):

if level < 1 or level > 4:

print("Invalid level")

elif mark < 0 or mark > 100:

print("Invalid mark")

elif level == 1 or level == 2:

if mark >= 85:

print("Distinction")

elif mark >= 75:

print("Merit")

elif mark >= 65:

print("Pass")

else:

print("Fail")

elif level == 3:

if mark >= 80:

print("Distinction")

elif mark >= 70:

print("Merit")

elif mark >= 60:

print("Pass")

else:

print("Fail")

elif level == 4:

if mark >= 70:

print("Distinction")

elif mark >= 60:

print("Merit")

elif mark >= 50:

print("Pass")

else:

print("Fail")

# main code

level = int(input("Please enter level: "))

mark = int(input("Please enter mark : "))

exam\_grade(level,mark)

## Solution 4.8 – Exam Average

# Name : 08AExamAverage

# Author : John Merchant

# Date : 12 Jul 2016

# Purpose : Exercise to calculate exam average mark

def average(number1,number2,number3):

if (number1 < 0 or number1 > 100

or number2 < 0 or number2 > 100

or number3 < 0 or number3 > 100):

print("Invalid mark")

else:

average\_mark = (number1 + number2 + number3)/ 3.0

if average\_mark >= 65:

print(average\_mark,"Pass")

else:

print(average\_mark,"Fail")

maths\_mark = int(input("Please input mark (0 to 100) for Maths : "))

english\_mark = int(input("Please input mark (0 to 100) for English: "))

ICT\_mark = int(input("Please input mark (0 to 100) for ICT : "))

average(maths\_mark,english\_mark,ICT\_mark)

# Name : 08BExamAverageFunction

# Author : John Merchant

# Date : 11 Jul 2016

# Purpose : Exercise to calculate exam average mark using function

def average(number1,number2,number3):

average = (number1 + number2 + number3)/ 3.0

return average

maths\_mark = int(input("Please input mark (0 to 100) for Maths : "))

english\_mark = int(input("Please input mark (0 to 100) for English: "))

ICT\_mark = int(input("Please input mark (0 to 100) for ICT : "))

if (maths\_mark < 0 or maths\_mark > 100

or english\_mark < 0 or english\_mark > 100

or ICT\_mark < 0 or ICT\_mark > 100):

print("Invalid mark")

else:

average\_mark = average(maths\_mark,english\_mark,ICT\_mark)

if average\_mark >= 65:

print(average\_mark,"Pass")

else:

print(average\_mark,"Fail")

## Solution 4.9 – Volume

# Name : 09AVolume

# Author : John Merchant

# Date : 12 Jul 2016

# Purpose : Exercise to calculate volume of box

def volume(number1,number2,number3):

if number1 < 0 or number2 < 0 or number3 < 0:

print("Invalid length")

else:

volume = number1 \* number2 \* number3

print("Volume:",volume)

width = float(input("Please input width : "))

depth = float(input("Please input depth : "))

height = float(input("Please input height: "))

volume(width,depth,height)

# Name : 09BVolumeFunction

# Author : John Merchant

# Date : 03 May 2016

# Purpose : Exercise to calculate volume of box using function

def volume(number1,number2,number3):

volume = number1 \* number2 \* number3

return volume

width = float(input("Please input width : "))

depth = float(input("Please input depth : "))

height = float(input("Please input height: "))

if width < 0 or depth < 0 or height < 0:

print("Invalid length")

else:

print("Volume:",volume(width,depth,height))

## Solution 4.10 – Pythagoras

# Name : 10Pythagoras

# Author : John Merchant

# Date : 12 Jul 2016

# Purpose : Exercise to create Pythagoras' Calculator

import math

def calc\_a(b,c):

a = math.sqrt(c\*\*2 - b\*\*2)

return a

def calc\_b(a,c):

b = math.sqrt(c\*\*2 - a\*\*2)

return b

def calc\_c(a,b):

c = math.sqrt(a\*\*2 + b\*\*2)

return c

menu\_option = 0

while True:

print("Menu:")

print("1 - Find the length of a given b and c")

print("2 - Find the length of b given a and c")

print("3 - Find the length of c given a and b")

print("9 - Exit")

menu\_option = int(input("Enter Option: "))

if menu\_option == 1:

b = float(input("Please enter length b: "))

c = float(input("Please enter length c: "))

print("length of a =",calc\_a(b,c))

elif menu\_option == 2:

a = float(input("Please enter length a: "))

c = float(input("Please enter length c: "))

print("length of b =",calc\_b(a,c))

elif menu\_option == 3:

a = float(input("Please enter length a: "))

b = float(input("Please enter length b: "))

print("length of c =",calc\_c(a,b))

elif menu\_option == 9:

break

else:

print("Invalid option selected")