QA Consulting

Phyton Exercise Book

TRAINING MATERIALS - EXERCISE BOOK

Contacts

Thomas.Knowles@qa.com

team.qac.all.trainers@qa.com

www.consulting.qa.com

Contents

Introduction – 3
Basic – 3
Hello World! – 3
Assignment – 3
Parameters – 3
Parameters/Operators – 3
Conditionals – 4
Conditionals 2 – 4
Recursion – 4
Lists – 4
Recursion/Lists – 4
Recursion/Lists – 5
User input – 5
Partial Functions. – 5
Intermediate – 6
Blackjack – 6
Unique Sum – 6
Too hot? – 6
Leap Year – 6
Paint Wizard – 7
Testing – 7
Working with Files – 7
Library/TDD/OOP – 8
Interface – 9
IO – 9
Advanced – 10
Prime Numbers 1 – 10
Prime Numbers 2 – 10
Strings – 10
Strings 2 – 10
Battleship – 11

Introduction

This document contains a range of exercises in order to help you get to grips with Python. You are not expected to complete all tasks, but a minimum is listed below. You may complete any other exercises in addition to improve your skills as it is not expected that you should complete the below as a first task unless you already have Python OOP experience

- Paint Wizard
- Testing
- Library/TDD/OOP

Basic

HELLO WORLD!

Output “Hello World!” to the console.

ASSIGNMENT

Store “Hello World!” in a variable, then output it to the console.

PARAMETERS

Create a method that accepts a string as a parameter, and then outputs that string to the console.

PARAMETERS/OPERATORS

Create a method that accepts two integers as an input, then returns an integer that is a sum of the two integers given.

CONDITIONALS

Modify your method from the Parameters/Operators task to accept another parameter, a Boolean, which if it is true, the method will return a sum of the two numbers, and if it is false it will return the multiplication of the two numbers.

CONDITIONALS 2

Modify your method from the Conditionals task to have another if statement that checks if one of the numbers is 0, if this is true then return the other non-0 number.

```
Input -> 1, 0 Return 1
Input -> 1, 2 Return 3
```

RECURSION

Create a for loop that will call and output the result of your method from the Conditionals 2 task 10 times, using the current iteration as one of the parameters you pass to it.

LISTS

Create a list with 10 integer values in it, then call and output the result of your method from Conditionals 2 with values that are stored in the array.

RECURSION/LISTS

Using your list that you created in the Lists task, create a loop that iterates through your list, outputting the values contained within it.

RECURSION/LISTS

Create a loop that populates a list with values, outputting them at each iteration. Then create another loop that iterates through the array, changing the values at each point to equal itself times 10, outputting them at each iteration.

Example Output

```
1,2,3,4…
10,20,30,40…
```

USER INPUT

Modify the previous task to take input from the user, taking an integer off of the user and using that integer to determine how large the array is going to be.

PARTIAL FUNCTIONS.

Create a partial number for doubling any number. Create another one for tripling any number.

Intermediate

BLACKJACK

Given 2 integer values greater than 0, return whichever value is closest to 21 without going over 21. If they both go over 21 then return 0

```
Input(18,21) -> 21
Input(20,18) -> 20
Input(22,22) -> 0
```

UNIQUE SUM

Given 3 integer values, return their sum. If one value is the same as another value, they do not count towards the sum. Aka only return the sum of unique numbers given.

```
Input(1,2,3) -> 6
Input (3,3,3) -> 0
Input (1,1,2) -> 2
```

TOO HOT?

Given an integer value and a Boolean value, temperature and isSummer, if temperature is between 60 and 90 (inclusive), unless its summer where the upper limit is 100 instead of 90. Return true if the temperature falls within the range, false otherwise.

LEAP YEAR

Given a year work out if it is a leap year or not.

Rule: A year is a leap year if it is divisible by 4, and either divisible by 400 or not divisible by 100.

PAINT WIZARD

Create a paint requirement wizard that will calculate which of the following three products, would be the cheapest to buy, for the room you are painting.

Work out which one wastes the least.

Work out which is the best choice for any room (Cheapest).

1. CheapoMax (20Litre) £19.99
   - This tin of paint will cover 10m^2 per Litre
2. AverageJoes (15 Litre) £17.99
   - This tin of paint will cover 11m^2 per Litre
3. DuluxourousPaints (10 Litre) £25
   - This tin of paint will cover 20m^2 per Litre

TESTING

Implement unit tests for the Paint Wizard task. Simulate normal inputs and test their outputs.

WORKING WITH FILES

Create a class representing a person with 3 attributes Name, Occupation, Age.

Create a list and populate it with 5 of these objects (Make up the values etc.).

Create a loop to iterate through the list, writing each object to one file (Think about how you format this).

Separately, create another list and populate it with the data in the file you just wrote too. (You're going to have to parse it back in the format you wrote it in).

LIBRARY/TDD/OOP

Create a library system with functionality to manage items within the library.
Expectations:

- Class diagram created and adhered to as much as possible.
- TDD Implemented.
- At least one Abstract Class must be implemented.
- Each item should have at least 1 additional attribute unique to itself.
- Naming conventions adhered too
- Commenting where necessary.

Items:
At least 3 of the following:

- Books
- Maps
- Government documents
- Media resources (Camera etc.)
- Newspapers
- Journals
- Magazines
- Dissertations
- Theses

Functions:
At least the following:

- Check out item
- Check in item
- Add item
- Remove item
- Update item
- Register person
- Delete person
- Update person

INTERFACE

Implement a user console interface into your library project.

IO

Have an option to write the current library contents (Any items currently in the library) to a file, and another option to load them from a file into the library.

Advanced

PRIME NUMBERS 1

Create an algorithm that determines how many prime numbers are between 1 and 3 million.

Extension – Have it finish running in under 2 minutes

PRIME NUMBERS 2

Create an algorithm that determines how many prime numbers are between 1 and 2 billion.

Extension – Have it finish running in under 3 minutes

STRINGS

Given two strings, write a program that efficiently finds the longest common subsequence.

STRINGS 2

Given two strings, write a program that outputs the shortest sequence of character insertions and deletions that turn one string into the other.

BATTLESHIP

Create the battleship game!

This project is to create a digital version of the popular board game known as battleships. Battleship is a two player game with 2 phases. In the first phase the player's ships are placed on the board. In the second phase each player takes it in turns to select grid squares on the board in an attempt to find and destroy their opponent's ships. Once one player has lost all of their ships the game is over and the player who still has ships on the board is the winner.

Each player has a number of ships including:

- 2 patrol boats (1 x 2)
- 2 battleships (1 x 3)
- 1 submarine (1 x 3)
- 1 destroyer (1 x 4)
- 1 carrier (1 x 5)

There are a number of rules that players must follow.

- 2 Ships cannot occupy the same space on the board.
- If a player scores a 'hit' on their opponent, then they get a second shot.
- Ships cannot be moved once they have been placed.

Advice
Battleships is a seemingly simple strategy game but without careful planning it can be easy to become "lost" in the project. It is recommended that you attempt to complete the project in a set of stages where with each stage you increase the level of complexity. Remember that as the complexity of your project increases you may find that you wish to go back to a previous version in some cases so it is highly advised that you create versioned copies of your project at each stage. An advised set of stages are:

Stage 1
A 3 x 3 grid with one ship that is 2 pieces long is placed on in the grid.

Stage 2
A 3 x 3 grid with 2 ships that are 2 pieces long and placed on the grid with validation to ensure legal placement.

Phyton

Stage 3
Two 3 x 3 grids with 2 ships where players take alternating turns taking shots at the other grid.

Stage 4
Differentiation between ‘hits’ and ‘misses’ implemented.

Stage 5
Checks for sunk ships with game over when one player has lost all their ships.

Stage 6
Two 12 x 12 grids with all 7 ships placed in valid locations.

Stage 7
Players can select the placement of their ships on the grid during phase 1.

Stage 8
Implementation of an AI player.

Stage 9
Players can only see their own grid with shots taken.

Stage 10
Implementation of a GUI has been attempted.