example_entry_test

January 25, 2021

1 Example Python Entry Test

1.1 MSc Health Data Science, University of Exeter Medical School

This is an **EXAMPLE** entry test to help you prepare for the real entry test. Questions in your actual test will vary.

The entry test is designed to evaluate your understanding of the basics of coding in Python 3.x

Please practice your python skills before taking the test. **You must be proficient** in the following syllubus:

- Python variables
- String manipulation
- Creating, manipulating, searching, sorting, filtering and modifying lists and items within them.
- Conditional `if-elif-else` statements
- Iteration, for example, `for`, `while` and list comprehensions.
- Creating functions that accept parameters and return individual variables and tuples.
- The basics of working with files and input/output.
- Coding basic mathematical formulas.
- importing and use of python built-in mathematical modules. For example, `math`,`random` and `itertools`

There are 7 questions worth a total of 30 marks. The pass mark for the test is 15 marks.

You are allowed to use any text editor or IDE. You are not allowed to use any other software, search the internet for answers or look at any othe reference material including a book.

Enjoy and good luck

1.1.1 QUESTION 1

[1 mark]

Write a python statement to display the sum of the values in the list:

```
x = [3, 5, 9, 1]
```

EXAMPLE ANSWER:

[1]:
```
x = [3, 5, 9, 1]
print(sum(x))
```

```
18
```

1.1.2 Question 2

[3 marks]

Present value (PV) is a financial calculation to find the current value of a future sum of money or cash stream in today's money at a specific rate of return.

Write a function to calculate the present value of a future value.

The **present value** of a future value is calculated as follows:

$$PV = \frac{FV}{(1+r)^n}$$

Where **r** is rate of return and **FV** is the future value.

Test case 1 (expected answer to 2dp = 1683.95)

```
future_value = 2000
rate = 0.035
n = 5
```

Test case 12 (expected answer to 2dp = 316.85)

```
future_value = 350
rate = 0.01
n = 10
```

EXAMPLE ANSWER

[2]:
```
def pv(future_value, rate, n):
    '''
    Discount a value at defined rate n time periods into the future.

    Forumula:
    PV = FV / (1 + r)^n
    Where
    FV = future value
    r = the interest rate
    n = number of years in the future

    Params:
    -------
    future value: float,
        The value to discount

    rate: float
```

```
        the rate at which to do the discounting

    n: float,
        the number of time periods into the future

    Returns:
    --------
    float
    '''
    return future_value / (1 + rate)**n
```

[3]:
```
#Test case 1
future_value = 2000
rate = 0.035
n = 5
result = pv(future_value, rate, n)

#results formatted to 2dp
print(f'{result:.2f}')

#Test case 2
future_value = 350
rate = 0.01
n = 10
result = pv(future_value, rate, n)

#results formatted to 2dp
print(f'{result:.2f}')
```

```
1683.95
316.85
```

1.1.3 QUESTION 3:

[5 marks]

Write a function called `fizzbuzz` that accepts an integer parameter `n`. The function should check if `n` is multiple of 3, 5 or both. The function should return a string as specified below

- If the number is a multiple of 3 return "FIZZ"
- Else If the number is a multiple of 5 return "BUZZ"
- Else If the number is a multiple of 3 AND 5 return "FIZZBUZZ"
- Else return the number (cast as a string)

Input data to test 1, 3, 5, 15, 23

Expected Output 1, "FIZZ", "BUZZ", "FIZZBUZZ", "23"

EXAMPLE ANSWER

[4]:
```
def fizzbuzz(n):
    """
    Fizzbuzz game.
    For multiples of three print "Fizz" instead of the number
    and for the multiples of five print "Buzz".
    For numbers which are multiples of both three and five print "FizzBuzz".

    Parameters:
    -----------
    n: int
        The integer number to test

    Returns:
    -------
    str
    """
    if n % 3 == 0 and n % 5 == 0:
        return 'FIZZBUZZ'
    elif n % 3 == 0:
        return 'FIZZ'
    elif n % 5 == 0:
        return 'BUZZ'
    else:
        return str(n)
```

[5]:
```
#test cases
print(fizzbuzz(1))
print(fizzbuzz(3))
print(fizzbuzz(5))
print(fizzbuzz(15))
print(fizzbuzz(23))
```

```
1
FIZZ
BUZZ
FIZZBUZZ
23
```

1.1.4 QUESTION 4

[4 MARKS]

The function `convert_celsius_to_fahrenheit()` provided below accepts a float parameter `deg_celsius` that represents a temperature in degrees celsius. Using the standard formula it calculates and returns a float representing the corresponding temperature in degrees fahrenheit.

The list below represents a range of temperatures in degrees celsius that a researcher needs to convert into degrees fahrenheit.

```
celsius = [39.2, 36.5, 37.3, 41.0]
```

Code a python programme that **creates a new list called** `degrees_f`**.** The code must **loop** over the `celsius` list and repeatedly call `convert_celsius_to_fahrenheit` passing in the current list item in the iteration. Print the new list `degrees_f` (temperature in degrees fahrenheit) to the screen.

FUNCTIONS PROVIDED

[6]:
```
def convert_celsius_to_fahrenheit(deg_celsius):
    """
    Convert degress celsius to fahrenheit
    Returns float value - temp in fahrenheit

    Parameters:
    -----------
    deg_celcius: float
        temp in degrees celsius

    Returns:
    -------
    float
    """
    return (9/5) * deg_celsius + 32
```

[7]:
```
#list of temps in degree celsius to convert to fahrenheit
celsius = [39.2, 36.5, 37.3, 41.0]
```

EXAMPLE ANSWER 1

[8]:
```
#solution using a for loop
degrees_f = []
for degrees_c in celsius:
    result = convert_celsius_to_fahrenheit(degrees_c)
    degrees_f.append(result)
print(degrees_f)
```

```
[102.56, 97.7, 99.14, 105.8]
```

EXAMPLE ANSWER 2

[9]:
```
#solution using a list comprehension
degrees_f = [convert_celsius_to_fahrenheit(degrees_c) for degrees_c in celsius]
print(degrees_f)
```

```
[102.56, 97.7, 99.14, 105.8]
```

1.1.5 QUESTION 5

[6 MARKS]

- Write a function that accepts a `List` of integers as a parameter and then finds and **returns** the maximum value in the list.

- Print the result to the screen.
- You should not use the built-in python `max` function.

Input data:

- Create a `List` of integer values

```
to_search = [0, 1000, 2, 999, 5, 100, 54]
```

Expected Output * 1000

[10]:
```
import math

def find_max(to_search):
    '''
    Simple iteration to find maximum value.
    Assumes all items in to_search are numeric.

    Parameters:
    -----------
    to_search: array-like
        a list of numeric values

    Returns: float
        Maximum value from list
    '''
    #alternative: set to a very small number e.g. -99_999
    current_max = -math.inf

    for item in to_search:
        if item > current_max:
            current_max = item

    return current_max
```

[11]:
```
#test case
to_search = [0, 1000, 2, 999, 5, 100, 54]
result = find_max(to_search)
print(f'Max value is {result}')
```

```
Max value is 1000
```

2 Question 6

[5 marks]

The variable `list_of_lists` represents a multi-dimensional array i.e. it is a list containing smaller lists.

```
list_of_lists = [[8, 2, 1], [9, 1, 2], [4, 5, 100]]
```

Code a python programme to **flatten** the list. The output of your programme should be a 1 dimensional list i.e.

```
result = [8, 2, 1, 9, 1, 2, 4, 5, 100]
```

EXAMPLE SOLUTION 1:

[12]:
```
#Solution using a nested iteration
list_of_lists = [[8, 2, 1], [9, 1, 2], [4, 5, 100]]

flat_list = []
for row in list_of_lists:
    for col in row:
        flat_list.append(col)

print(flat_list)
```

```
[8, 2, 1, 9, 1, 2, 4, 5, 100]
```

EXAMPLE SOLUTION 2:

[13]:
```
#solution using a list comprehension
list_of_lists = [[8, 2, 1], [9, 1, 2], [4, 5, 100]]
flat_list = [item for sublist in list_of_lists for item in sublist]

print(flat_list)
```

```
[8, 2, 1, 9, 1, 2, 4, 5, 100]
```

EXAMPLE SOLUTION 3:

[14]:
```
#solution using itertools
import itertools

list_of_lists = [[8, 2, 1], [9, 1, 2], [4, 5, 100]]
flat_list = list(itertools.chain(*list_of_lists))

print(flat_list)
```

```
[8, 2, 1, 9, 1, 2, 4, 5, 100]
```

2.1 Question 7

[6 marks]

You are given a list of comics:

```
comics = ['Iron-man', 'Captain America', 'Spider-man', 'Thor', 'Deadpool']
```

Tasks:

- slice and then print the first and second list items
- slices and the print the second to fourth list items
- slice and then print the fourth and fifth list items
- append "Doctor Strange" to the list. Print the updated list
- insert "Headpool" before "Deadpool" in the list. Print the updated list
- delete "Iron-man". Print the updated list

EXAMPLE SOLUTION

[15]:
```
comics = ['Iron-man', 'Captain America', 'Spider-man', 'Thor', 'Deadpool']

first_two = comics[:2]          # slice the first and second list items
middle_three = comics[1:4]      # slices the second to fourth list items
last_two = comics[-2:]          # slice the fourth and fith list items

#print answers
print(first_two)
print(middle_three)
print(last_two)

comics.append('Doctor Strange')   # append 'Doctor Strange' to the list
comics.insert(4, 'Headpool')      # insert 'Headpool' before 'Deadpool'
del comics[0]                     # delete #'Iron-man'
print(comics)                     # print the updated list.
```

```
['Iron-man', 'Captain America']
['Captain America', 'Spider-man', 'Thor']
['Thor', 'Deadpool']
['Captain America', 'Spider-man', 'Thor', 'Headpool', 'Deadpool', 'Doctor
Strange']
```