

# Assignment 2: Python Basics and GitHub Pages Website

For this assignment, see the QMD or IPYNB files in our course repository named 02-python\_basics.qmd or 02-python\_basics.ipynb.

Most things in this can be solved using methods shown in lecture. However, some cases will require you to search the internet for answers.

## Question 1

a). **Indentation and Pythonic Code:** - Create a function named `print_numbers` that prints numbers from 1 to 10, each on a new line. Ensure your code is properly indented and follows Pythonic conventions.

```
#1 a) Answer
def print_numbers():
    for i in range(1, 11):
        print(i)

print_numbers()
```

```
1
2
3
4
5
6
7
8
9
10
```

b.) **Rules of Functions and Comments:** - Define a function named `calculate_area` that takes two arguments `length` and `width`, and returns the area of a rectangle. Include a docstring explaining what the function does. See the lecture notes for an example of a docstring.

```
#1 b) Answer:
def calculate_area(length, width):
    """Returns the area of a rectangle"""
    rectangle_area = length * width
    return rectangle_area

# Example
rectangle = calculate_area(5, 3)
print(rectangle)
```

15

c.) **Basic Data Types:** - Create a variable `is_active` and set it to `True`. Print the type of `is_active` using the `type()` function.

```
#1 c) Answer:
is_active = True

print(type(is_active))
```

<class 'bool'>

d.) **Working with Lists:** - Create a list named `fruits` containing some fruit names. Use the `append` method to add 'Mango' to the list and print the updated list.

```
#1 d) Answer:
fruits = ["apple", "watermelon", "banana", "cantoloupe"]

fruits.append("mango")

print(fruits)
```

['apple', 'watermelon', 'banana', 'cantoloupe', 'mango']

## Question 2:

a.) **Naming Conventions:** - Create a variable following the snake\_case naming convention to store the number of days in a week.

#2 a) Answer:

```
number_days_in_week = 7
```

b.) **Control Flows - Conditionals:** - Write a simple program that checks if a number is positive, negative, or zero using if, elif, and else statements.

#2 b) Answer:

```
number = 0.45 #example value

if number > 0:
    number_check = "positive"
elif number == 0:
    number_check = "zero"
else:
    number_check = "negative"

print(f"The number is {number_check}")
```

The number is positive

c.) **Control Flows - Loops:** - Write a for loop to calculate the sum of numbers from 1 to 100.

#2 c) Answer:

```
total = 0
for i in range(1, 101):
    total += i

print(total)
```

5050

d.) **Control Flows - Loop Control Statements:** - Modify the loop from the previous sub-question to stop the loop if the sum becomes greater than 1000 using a **break** statement.

```
#2 d) Answer:
total = 0
for i in range(1, 101):
    total += i
    if total > 1000:
        break #breaks at 45, because this eclipses 1000
print(total)
```

1035

e.) **Functions - Arguments and Return Values:** - Create a function named `get_full_name` that takes two arguments `first_name` and `last_name`, and returns the full name.

```
#2 e) Answer:

def get_full_name(first_name, last_name):
    full_name = f"{first_name} {last_name}"
    return(full_name)

#Example
print(get_full_name("Henry", "Kombol"))
```

Henry Kombol

### Question 3

**Part A:** Write a python function that calculates the sum of all squared numbers from 1 to x. Illustrate that it works for  $x = 20$ .

HINT, `**` is the exponent operator in python. HINT syntax for a python function is:

`def function_name(variable_name): outcome = variable_name + 1 return outcome`

```
#3A Answer
def squared_calc(x):
    num_sum = 0
    for i in range(1, x + 1):
```

```

    num_sum += i**2
    return(num_sum)

print(squared_calc(20))

```

2870

The python library named “os” is a built-in library for dealing with any file on your operating system. Often, research tasks involve LOTS of files and you need to iterate over them. To show you know how to do this, use the `os.listdir` function to answer the following questions. Note that you need to write “import os” to import the library into your code before you can use it:

**Part B:** Print out a list of all the files your class root directory. I don’t care how many are actually there (in case you’ve added some yourself) but show me how.

```

#3B Answer
import os

files = os.listdir('/Users/henrykombol/Desktop/apec_8222/apec_8222_henry')
print(files)

```

```
['python_basics', 'LICENSE.md', 'spatial_analysis', 'assignments', '.DS_Store', 'environment']
```

**Part C:** Using a FOR loop, iterate over the list from above. Using the function `len()`, count how many letters there are in the filenames. HINT just like in real life, you may need to google the `len()` function and see how it work on e.g. strings.

```

#3C Answer
for i in files:
    print(len(i))

```

```

13
10
16
11
9
15
16
9
10
4

```

**Part D:** Write a Python program which iterates over the integers from 1 to 50. 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”. Hint: look up how to use the modulo operator (which is written as % in python) which would let you test when the remainder of a division is exactly 0.

Side-note, this is a hilariously over-used question that most software engineers get asked on their first interview for a job.

```
# 3D Answer
for i in range(1, 51):
    if i % 5 == 0 and i % 3 == 0:
        print("FizzBuzz")
    elif i % 5 == 0:
        print("Buzz")
    elif i % 3 == 0:
        print("Fizz")
    else:
        print(i)
```

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
16
17
Fizz
19
Buzz
Fizz
22
23
```

Fizz  
Buzz  
26  
Fizz  
28  
29  
FizzBuzz  
31  
32  
Fizz  
34  
Buzz  
Fizz  
37  
38  
Fizz  
Buzz  
41  
Fizz  
43  
44  
FizzBuzz  
46  
47  
Fizz  
49  
Buzz

#### Question 4

Create a personal website using GitHub Pages. The website should be just a few pages, including a landing page, a research page, and a contact page. Use Markdown to format the content on your pages. Include links to your GitHub profile and any relevant projects or repositories you have worked on. Once completed, share the link to your GitHub Pages website here. Note, for the pages to actually be hosted online, you will need to create a new repository named `.github.io` and push your quarto-rendered markdown files there. See GitHub Pages documentation for more details.

github website link: <https://henryk3100.github.io/>

### Hint 1

Organization and customization of your site will be managed by creating a `__quarto.yml` in your project root with content like below.

```
project: type: website
```

```
website: title: "Your Name" navbar: right: - text: "Home" href: index.qmd - about.qmd - projects.qmd - contact.qmd
```

```
format: html: theme: cosmo toc: true code-copy: true
```

### Hint 2

Individual pages could be written as, for example, a `index.qmd` with the following content.

## Hello, I'm Your Name

Short tagline (e.g., *Applied Economist • Data Scientist*).

- I study ...
- I teach ...
- I build ...

### Hint 3

Call `quarto render` on the command line in your project root to build the site and generate HTML files.

### Hint 4

To actually host the site, push the rendered HTML files to a GitHub repository named `.github.io`. Follow GitHub Pages documentation for details on setting up the repository and enabling GitHub Pages.

Ideally, this should be a public site. However if you would like to remain anonymous, you can just create a private repository and invite me as a collaborator on the repository.