

# ENGG1811 Week 10 Lab

## Instructions

- (a) This lab consists of 2 questions.
- (b) Time allowed: 50 minutes.
- (c) For each question, an associated test file or testing code has been provided to help you to test your code. It is your responsibility to test your code thoroughly before submission. You may add additional tests if you wish.
- (d) Each test file has multiple test cases. You should aim to pass as many test cases as you can.
- (e) Each question requires you to submit a separate Python program file for marking.
  - (i) You will be provided a submission link for each question.
  - (ii) Each question requires a specific filename and the submission system will only accept that particular filename.
  - (iii) Ensure that you **save** your file before submission.
  - (iv) If the submission system accepts your file, it will also display the contents of your submitted file and runs the same tests in the test file.
  - (v) You can submit at any time during the lab. There is a grace period at the end of the 50-minute to allow you to make your final submissions.
  - (vi) We will only mark the last submission that you made.
- (f) You are allowed to consult lecture notes, books and materials on the web.
- (g) The lab will be un-supervised. You are expected to do the lab yourselves. No communications are allowed.

## Question 1

Bush walkers use a simple equation, called Naismith's Rule, to estimate the amount of time taken in hours for a walk given its distance  $d$  in kilometres, climb  $h$  (number of metres of height gained over all uphill sections) and the average walking speed  $s$  in km/hr on flat parts of the walk:

$$t = \frac{d}{s} + \frac{h}{400}$$

For example, a 21km walk with a 600m climb undertaken at an average speed of 3.5 km/hr should take about  $21/3.5 + 600/400 = 6.0 + 1.5 = 7.5$  hours.

A walk is classified as **Easy** if it takes less than 4 hours and also the climb is less than 200m; it is **Hard** either if it takes more than 8 hours or the climb is more than 600m; and it is **Medium** in any other case.

Your task is to write a Python function whose `def` line is:

```
def q1_func(d, h, s):
```

The three function inputs `d`, `h` and `s` correspond to the symbols  $d$ ,  $h$  and  $s$  that were used earlier. The function is expected to return a string as its output.

You can assume that when we call the function `q1_func`, the supplied inputs `d`, `h` and `s` are always numbers of either `float` or `int` type. The supplied values of `d`, `h` and `s` can be positive, zero or negative. For the parameters `d` and `s` (for distance and speed) to be valid, both of them must be strictly positive, i.e. cannot be zero. For the parameter `h` (for height) to be valid, it must be greater than or equal zero, i.e. zero is allowed for height.

The function `q1_func` is expected to do the following:

- Step 1 Check if all supplied values are valid. If yes, then proceed to Step 2; otherwise, it should return the string `Invalid` and it should not proceed to Step 2.
- Step 2 Use the given `d`, `h` and `s` to classify the walk. Depending on the classification, the function should return one of the strings: `Easy`, `Medium` or `Hard`.

### Requirements and testing:

- You must write the function `q1_func` in a file with the filename `q1.py`. The submission system will only accept this filename. A template file `q1.py` has been provided.
- You can use the file `test_q1.py` for testing.
- You do not need to submit `test_q1.py`.
- Make sure that you **save** your file before submission.

## Question 2

Note that if this question appears in the exam, a differential marking scheme may be used. For a differential marking scheme, you can only receive the maximum mark if your solution is correct and it does not use any loops, while a correct solution that uses loops will receive only a fraction (possibly 70%) of the maximum. However, for this lab, a differential marking scheme will **not** be used but you are encouraged to try to solve this problem without using loops.

Your task is to write a Python function `q2_func` with the following `def` line:

```
def q2_func(x, y):
```

The function inputs `x` and `y` are assumed to be 1-dimensional numpy arrays of the same shape. The function is required to compute and return a numpy array which has the same shape as both `x` and `y`. In the following description, we will refer to the numpy array to be returned by the variable name `z`.

Let `x[i]`, `y[i]` and `z[i]` be the element indexed by `i` in, respectively, the arrays `x`, `y` and `z`. The relationship between `x[i]`, `y[i]` and `z[i]` is given by the following pseduo-code:

```
if (absolute value of x[i] > absolute value of y[i])
    z[i] = x[i] - y[i]/2
else
    z[i] = y[i] - x[i]/2
```

For examples:

- If `x[i]` is 4 and `y[i]` is 3, then `z[i]` should take on the value of  $4 - 3/2 = 2.5$ .
- If `x[i]` is 3 and `y[i]` is -4, then `z[i]` should take on the value of  $-4 - 3/2 = -5.5$ .

### Requirements and testing:

- You must write the function `q2_func` in a file with the filename `q2.py`. The submission system will only accept this filename. A template file `q2.py` has been provided.
- Your function must be able to work with any two 1-dimensional numpy arrays `x` and `y` of the same shape. You can assume that we will only test your function with a pair of arrays of the same shape.
- You can use the file `test_q2.py` for testing.
- You do not need to submit `test_q2.py`.
- Make sure that you **save** your file before submission.