

# JC1001 Assessment 2 Instructions

## General information

- **Please read and follow all instructions carefully**
- This assignment is worth 25% of your total grade for this course.
- The deadline is 23:00 19/12/2025 SCNU local time
- All normal assignment regulations apply including penalties for late submissions and plagiarism
- **This is an individual assignment and the work submitted must be your own and will be checked to ensure the code you have submitted is based on your own individual effort**
- Please see the student handbook for further information.
- Please note that there are three parts of this assignment. Create a Python file and name it with your **University of Aberdeen ID number and extension py (50XXXXXXX.py)**. All the functions for this Assessment should be saved into that single file. **Make sure you name your functions correctly based on the instructions in this Assessment brief.**
- A copy of the function specification for each question is included in this document
- To open the assessment brief, click at the Assessment 2 folder.

# About the assignment

There are 16 functions you are required to write solutions for in the `ca2_submission` file, open in the right panel. **Write all solutions for each function in the `ca2_submission` file, using the exact functions names as specified in the function specifications.**

## Part 1 instructions

In this part of the assignment, you will write 6 functions related to calculating and outputting various information related to Body Mass Index (BMI) measurements. The total number of marks for each question is listed after each function specification.

### Question 1

---

Write a function named ***stringToNum*** that receives a string parameter of numbers separated with commas and returns a list of integers based on the string parameter e.g. ***stringToNum("5,6,7,8")*** would return ***[5,6,7,8]***.

(2 marks)

### Question 2

---

Write a function named ***n\_w\_h\_input*** that receives a string followed by two floats. The function should return a tuple containing the string argument in upper case and the two floats rounded to 1 decimal place.

(2 marks)

### Question 3

---

Write a function named ***n\_w\_h\_output*** that receives a string followed by two floats. The function should print out these parameters using the format below e.g. if the function is called with the values, ***Sam,69.7,1.9*** the function should print:

“Sam’s weight is 69.7 and his/her height is 1.9”

(2 marks)

### Question 4

---

Write a function named ***calcBMI*** which receives two floats representing weight and height and then calculates body mass index (BMI). Your function should use the following formula:

BMI is calculated as:  $\text{weight} / \text{height} / \text{height} * 10,000$

The number should be rounded to 1 decimal place.

(2 marks)

### Question 5

---

Write a function named ***bmiCat*** which receives a float representing BMI and then returns a weight category based on the following criteria:

BMI categories are as follows:

Lower than 18.5, Underweight

Between 18.5 and 24.9, Healthy

Between 25 and 29.9, Overweight

Between 30 and 39.9, Obese

More than 40, Severely obese

(10 marks)

## Question 6

---

Write a function named **bmiReport** which receives a name, weight, height, bmi and weight category. The function should return a nested dictionary. The outer dictionary should contain a single key, that is the name passed as a parameter. The inner dictionary should contain keys for each of the remaining parameters, weight, height, BMI, weight category and each key should be assigned the value passed when the function is called.

(2 marks)

## Part 2 instructions

In this part of the assignment, you will write 3 functions related to performing various operations with lists and strings.

## Question 7

---

Write a function named **oddList**, which receives two integers and returns a list of all odd numbers between the first and second integer parameters. The returned list should include the second integer, if it is an odd number.

(4 marks)

## Question 8

---

Write a function named **reverseString**, which receives a string and returns the string with all characters reversed e.g. if the function is called with the parameter `CAT` it will return `TAC`.

(2 marks)

## Question 9

---

Write a function named **startAndEnd**, which receives a list and returns True if the start and end of the list are the same e.g. if the function is called with the parameter **[1,2,1]** it will return true but if the function is called with the parameter **[1,2,3]** it will return False.

(4 marks)

## Part 3 instructions

For the next part of this assignment, you will write several functions to implement a simple version of the game noughts and crosses (also known as tic-tac-toe). This is a two-player game which uses a grid of 9 squares. The grid is commonly referred to as a board. Each player is represented as either an X or a O symbol and each take turns to insert their symbol on the board. The game is won when either player X or player O successfully enters 3 of their symbols in a line horizontally, vertically, or diagonally. The game is a draw if neither player can win because there is no means by which either player can create a horizontal, vertical, or diagonal line of their respected symbols.

## Question 10

---

Write a function called **createBoard**. The function receives no parameters but will return a list containing 3 separate individual lists that represent the noughts and crosses board. Within each list, there should be 3 items containing the string “\_”.

E.g. a single list will contain **["\_","\_","\_"]** and there should be 3 to represent all 9 squares of the noughts and crosses board.

(2 marks)

## Question 11

---

Write a function named ***displayBoard***. The function should receive a single parameter, that is a list containing 3 lists, as described in (10). Your function should display each of the 3 lists on a single line. E.g.

```
[" _ " " _ " " _ "]
[" _ " " _ " " _ "]
[" _ " " _ " " _ "]
```

(2 marks)

## Question 12

---

Write a function named ***getMove***. This function receives no parameters but prompts the user to enter a number between 1 to 9. The function should return the number entered by the user or if the user does not enter a number between 1 to 9, the function should return False.

(4 marks)

## Question 13

---

Write a function named ***intToBoard***. The function should receive a single integer and then return a tuple that contains two integers that represent the coordinates of a single square from the noughts and crosses board. E.g. Each square on the board may be numbered as follows:

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

Square 1 may be accessed using (0,0). Therefore, if ***intToBoard*** receives 1 as a parameter, it should return the tuple (0,0). Note, consider the number of squares (9) and use of the floor and modulo operators.

(26 marks)

## Question 14

---

Write a function named ***insertToBoard***. This function should receive 3 parameters; the first parameter should be a Tuple, similar to that returned from ***intToBoard***. The second parameter should be a list of 3 lists i.e. the board and the final parameter should be a Boolean value (either True or False). The ***insertToBoard*** function should as name implies, insert a players move on the board. If the Boolean parameter is True, then the function should insert an “X” into the position of the board, provided by the Tuple of the first parameter. The function should also return a tuple with the value True and the board (list of 3 lists). If the Boolean parameter is False, then the function should insert an “O” in the board and return a tuple with the value True and the board. The board should only be updated with valid moves, that is areas on the board which do not contain a “X” or “O”. If the position specified by the first parameter contains a previous move i.e. an “X” or “O” then the function should return False and the board.

Remember, a tuple’s items may be accessed similarly to a list: i.e. consider the Tuple named t, which contains 3 elements: **t = (1,2,3)**. To access the first element, we would use **t[0]**.

(6 marks)

## Question 15

---

Write a function named ***checkDraw***. The function receives a single parameter, the board (list of 3 lists). The function should check whether there are any free squares left on the board i.e. a square containing “\_”. If there are squares on the board containing free spaces, then the function should return False, otherwise it should return True.

(10 marks)

## Question 16

---

Write a function named ***checkWin***. The function receives a single parameter, the board (list of 3 lists). The function should check whether any one of the 8 possible winning conditions are present. The winning conditions are when a horizontal, vertical or diagonal line of 3 squares on the board contain the same symbol (“X” or “O”). If either of these conditions are met, the function should return True, otherwise it should return False.

(20 marks)