University of London

Assessment Coversheet

Complete this coversheet and read the instructions below carefully.

**Candidate Number**: WP1242
Refer to your Admission Notice

**Degree Title**:
BSc in Computer Science

**Course/Module Title**:
Programming with Data

**Course/Module Code:**
CM2015

**Enter the numbers, and sub-sections, of the questions in the order in which you have attempted them:**

**Section A**
Online

**Section B**
Question 1
a, b, c, d, e, f, g, h, i

Question 3
a, b, c, d, e, f, g, h

**Date**: 19 March 2021

## Instructions to Candidates

1. Complete this coversheet and begin typing your answers on the page below, or, submit the coversheet with your handwritten answers (where handwritten answers are permitted or required as part of your online timed assessment).
2. Clearly state the question number, and any sub-sections, at the beginning of each answer and also note them in the space provided above.
3. For typed answers, use a plain font such as Arial or Calibri and font size 11 or larger.

4. Where permission has been given in advance, handwritten answers (including diagrams or mathematical formulae) must be done on light coloured paper using blue or black ink.
5. Reference your diagrams in your typed answers. Label diagrams clearly.

**The Examiners will attach great importance to legibility, accuracy and clarity of expression.**

**Begin your answers on this page**

## SECTION B

### Question 1

a)

The open function opens a file for use in a program.

The first parameter is the path and name of the file. This is the file to be opened. The second parameter is the mode in which the file will be opened. Valid modes include reading (“r”), writing (“w”), appending (“a”) and creating (“c”) with modifiers “t” and “b” to open files in textual or binary mode. An example of the two parameters is shown below.

```
open(“source_data.csv”, “r”)
```

b)

One of three numeric types is `int` for integer numbers. These are arbitrary precision signed integers only. Example: 12894

A second type is float for double-precision (64-bit) floating-point numbers. Example: 7.392

c)

An object is mutable when the objects or values contained in it can be modified. An immutable object is an object whose contents (other objects or values) cannot be modified.

d)

Example 1: List. A list can be modified, for example by adding or removing values.

Example 2: Dict. A dictionary can be modified, for example by changing the key-value pairs it contains.

e)

Example 1: Strings. A string cannot be modified, and the string characters are immutable.

Example 2: Tuples. Tuples cannot be modified once defined, i.e., their constituent parts are fixed and can only be modified by replacing the entire tuple with a new tuple.

f)

Associative mapping in the “Dict” data type describes the association of each key with a value. It means that every element of a dict is an association of a key and its value. It is useful for storing properties of some logical object, such as structured information about a person. Example:

```
my_dict = {‘name’: ‘John Doe’, ‘address’: ‘Apple Street 2’,
‘city’: ‘London’}
```

g)

A for loop based on an iterator iterates over every element in a collection (such as a list or a dict) and executes a piece of code once for that element. An example of an iterator-based for loop, assuming my_list is an iterator.

```
for item in my_list:
   print(item)
```

The sequence is:

1. The elements of **my_list** are unpacked and the first element is assigned as the value of **item**
2. The body of the loop is executed. In this case, the current value of item is printed.
3. The next element of **my_list** is now assigned to **item**, and the loop continues until no more elements are available in **my_list**.

h)

An iterable is an object that can be iterated over. It is not necessarily an iterator. In a for loop, this means that the object to be iterated over need to provide an iterator. The sequence is:

1. The for loop calls the iter() function on the iterable. If it is successful, the iterator passed back is used to iterate over the elements of the iterable object.

2. At the end of each execution of the for loop's body, the __next__() function is called to move the iterator forward by one.
3. The iteration ends when the __next__() function doesn't return an object.

i)

i. The questions I was trying to answer were related to the efficiency of betting markets in the English premier league. Specifically:
   - How often were betting markets wrong about the predicted result?
   - How has the error rate, defined as betting odds not matching the actual outcomes, evolved over the years; Have betting markets become better at predicting the actual winners over time?
   - Which team is the most predictable & unpredictable team?

ii. The dataset was a set of multiple CSV files containing all English Premier League matches of the past 10 years as records. Each file contained one year of data. Each record (line) in the CSVs represents one match, with the betting odds from various bookkeepers, the match result and some additional data about the matches. The dataset was obtained from a public data website (Datahub).

iii. The process taken for analysis was as follows:
   - **Data Preparation**: I imported all CSV files and combined them into a single data frame.
   - **Reducing the data**: I dropped all unneeded columns from the dataset which were irrelevant to the analysis
   - **Normalizing data**: Since the data was from different files over many years, I needed to make sure the data formats were consistent across all files. Especially date formats needed to be harmonized.
   - **Extending the data**: I added columns useful for the analysis, such as the winner of a match (which wasn't present in the source data, and had to be inferred based on the match result and home/away configuration of a match).
   - **Data Analysis**: I analysed if a match was predicted correctly by the betting odds and added this data as a column. I calculated the overall rate of successful predictions in the data. I plotted the results
   - **Data Analysis II**: I analyzed the rate of successful predictions over time and visualized this as well to see if predictions improved in 10 years.
   - **Data Analysis III**: I analyzed the same data set by team, to see if prediction rates differed based on the teams in a match. This identified the most predictable and unpredictable teams.

**Question 3**

a)

The following list states these five characteristics.

1. A pandas DataFrame represents a rectangular table of data and contains an ordered collection of columns. Each column can be of different value type, and the DataFrame has both row and column indices.

2. A DataFrame can be constructed from fundamental Python data structures, such as dicts, lists or even NumPy arrays. For example:

   ```
   data = {‘state’: [‘Ohio’, ‘Ohio’, ‘Ohio’, ‘Nevada’],
   ‘year’: [‘2018’,’2019’,’2020’,’2021’], ‘pop’: [1.5, 1.7,
   1.9, 1.2]}
   ```

3. Pandas DataFrames can be re-indexed to conform to a new index using the **reindex** method. This allows changing the column or row indices of a DataFrame easily.

4. DataFrames can contain missing values which can be filled using the **ffill** method – Pandas then intelligently fills in missing data using a specified method which could be numerical interpolation or forward-filling.

5. DataFrames have methods to easily compute descriptive statistics about the data. For example, the **describe** method automatically outputs some basic statistical values such as the count of values, mean, standard deviation and so on.

b)

df.dropna() will remove missing data. By default, this function will remove **rows** where **any** data attribute is missing. It will return a copy of the dataframe by default, and not do the removal in-place.

c)

Real-world data often has incomplete or missing records. It is common that records are incomplete as data might go back many years, when not all attributes of a certain dataset were collected, or rely on human entry where some data might have gone missing, or data that was transferred between systems and got lost in the migration process.

d) An example of such data might be personal information where birth dates are involved. In the following dataset, the second row would be removed from the dataset using dropna().

| First Name | Last Name | Birth Date |
|---|---|---|
| John | Smith | 01.12.1985 |
| Jane | Doe | NA |
| Matt | King | 05.06.1972 |

e)

The tail() function provides the last 5 rows. Example code, assuming “df” is a dataframe:

```
df = pd.DataFrame({'food': ['burger', 'fries', 'pizza',
'shake', 'salad', 'chicken wings']})
df.tail()
```

f)

We can use the fillna() function in combination with the mean() function like so:

```
df.fillna(df.mean())
```

g)

The function describe() will provide summary statistics, like so:

```
df.describe()
```

It provides statistics such as the mean, standard deviation, counts and other values.

h)

DataFrame.iloc is used to select the data from a dataframe at specific, numerical index. Essentially, it selects a row in the tabular data by it’s row number.