Foundation

Introduction to Coding
Using Python

Exam paper 1

Time allowed: 90 minutes

FDM

About the exam

1. Structure

The exam contains 5 questions. Each question requires you to write a function which takes arguments and then returns a value.

You will have 90 minutes to complete the questions.

2 Marking criteria

Each question will be marked according to the following criteria:

- Does it work? This will be checked by running a series of test cases. All tests must pass to receive the marks. No partial marks will be given. You will be given a full set of test cases so that you can check your code before submitting it. Tests passed by use of hard coding will be deemed to have failed.
- Is it written using clean code?

3 Clean code

- Naming standards:
  - All variables start with a lower case letter and use camel case.
  - Clear descriptive variable names with no ambiguity. E.g. a float containing a price should be called 'price' but a list containing some prices should be called 'prices'
  - The function name is identical to the name specified in the question. If it isn't it will fail the tests.
- Spacing and indentation.
- No excessive use of comments
- No obsolete lines of code:
  - So no print statements in the function
  - No code which has no impact on the return value

- No input statements in the function
- No commented out lines of code
- No excessively complex code. Think about breaking code down into smaller, simpler functions if it gets too complex.

4 Marking allocation

- Question 1 (25%)
  - Passes all tests (22%)
  - Clean code (3%)
- Question 2 (25%)
  - Passes all tests (22%)
  - Clean code (3%)
- Question 3 (25%)
  - Passes all tests (22%)
  - Clean code (3%)
- Question 4 (12 marks)
  - Passes all tests (10%)
  - Clean code (2%)
- Question 5 (13 marks)
  - Passes all tests (10%)
  - Clean code (3%)

- Pass (75-79%)
  - Questions 1, 2 & 3 with clean code OR
  - Questions 1, 2, 3 & 4 without clean code OR
  - Questions 1, 2, 3 & 5 without clean code
- Merit (80-89%)
  - Questions 1, 2, 3 & 4 with clean code OR
  - Questions 1, 2, 3, 4 & 5 without clean code
- Distinction (90% +)
  - All questions with clean code

The questions

Question 1 - Minimum wage

In file `q1.py` write a function called `minimumWage` which will return the minimum wage for someone based on their age.

Argument:

- An integer (containing the person's age)

Return value:

- A float (containing the minimum wage for a person of that age)

Note: If age is below 12 or above 80 then return: `None`

The minimum wage for each age group is:

Age	Wage
Under 18	4.35
18-20	6.15
21-24	7.70
25+	8.21

Question 2 - Count occurrences

In file `q2.py` write a function called `numberOfOccurrencesInString` which will return the number of times a number (i.e 0 or 1 or 2 or 3 or 4 or 5 or 6 or 7 or 8 or 9) occurs in a string.

Arguments:

- A string

Return value:

- An integer (the number of times a number occurs in the string)

For example:

- String "`Commodore 64`" would return `2`
- String "`ZX Spectrum`" would return `0`
- String "`IBM PC 5150`" would return `4`

Question 3 - Cheaper prices

In file `q3.py` write a function called `percentageUnderTenPounds` which takes as the input a list of prices, and returns the percentage of prices which are under 10 pounds.

Arguments:

- A list of floats (containing the prices)

Return value:

- A single float (containing the percentage of prices which are under £10)

For example:

- List: `[5.99, 15.49, 25.00, 10.25, 15.50, 19.99]`
  would return `16.666` (1 out of 6 prices)

- List: `[2.50, 75.99, 39.50, 7.99]`
  would return `50.0` (2 out of 4 prices)

Question 4 - Crossword line

In file `q4.py` write a function called `fillCrosswordLine` which takes two strings as arguments.

Arguments:

The first string represents a **line** in a crossword. It has characters where dashes (-) represent a space for a word to be placed and pluses (+) represent places where a word can't be placed.

For example:

`+------++`	this string can fit a 6 letter word.
`+++----++`	this string can fit a 4 letter word.
`---++++`	this string can fit a 3 letter word.

The second string represents a **word** to fit into the space within the first string.

For example:

The word “help” would fit into this String: “++----++” but wouldn’t fit into “++------++” (too short) or “---+++++” (too long).

Return value:

- A string containing the completed crossword line

  Or None if the word doesn’t fit.

For example:

```
+++----++    and “help” should return “++++help++”
+------+     and “help” should return None
---++++      and “help” should return None
```

Please take note of the following:

- The strings can be any length
- The first string doesn’t have to start with a + and/or end with a +

Question 5 - Longest palindrome

In file q5.py write a function called longestPalindrome which checks a string for substrings which are palindromes and returns an integer containing the length of the longest palindrome.

A palindrome is a word, phrase, or sequence that reads the same backwards as well as forwards.

Argument:

- A string containing letters (ignore the case)

Return value:

- An integer containing the length of the longest substring which is a palindrome.
- If there are no palindromes contained within the string the function should return `0` (zero)

Note: A For the purpose of this exercise – a palindrome can NOT be a single letter.

Also do NOT ignore spaces – but you can ignore the case. MOM is the same as mom

For example:

- The string "abcd" has the following substrings: "`a`", "`ab`", "`abc`", "`abcd`", "`b`", "`bc`", "`bcd`", "`c`", "`cd`" and "`d`". A single letter cannot be a palindrome and none of the other stings read the same backward as forwards. So the function should return 0.
- The longest palindrome in the string "`annahasaracecar`" is "`racecar`" which has `7` characters making it longer than another substring "`anna`" which only has `4` characters.
- There are no palindromes in the string "`palindrome`". So return `0`
- In "`madam and eve`" – "`madam`" is a palindrome so return the length `5`
- "`nurses run`" reversed is "`nur sesrun`" which is not palindrome because the space is in the wrong place. But the sequence "`ses`" is a palindrome so return length `3`
- "`a`" is not a palindrome as it is a single letter so return `0`