Question 1. [5 marks]

For each code fragment in the table below, select the answer that best describes the printed output, or the error that occurs when the code is run.

```
diff = float(2) - int(str(2))
print(diff != 0)
```

(A) `True`
(B) `False`
(C) an error occurs converting one of the arguments

```
phrase = ['All', 'the', 'chocolate']
phrase.insert(3, 'now') + ['please']
print(phrase)
```

(A) `['All', 'the', 'chocolate']`
(B) `['All', 'the', 'chocolate', now']`
(C) `['All', 'the', 'chocolate', 'please']`
(D) `['All', 'the', 'chocolate', 'now', 'please']`
(E) some other list is printed
(F) an error occurs because + is not defined for lists
(G) an error occurs because + is not defined for non-list and list

---

```
print('CAT!'.isupper() or 1 / 0 == 5)
```

(A) `True`
(B) `False`
(C) a `ZeroDivisionError` occurs
(D) another error occurs

```
L = ['for', 'this', 'was', 'on',
    'seynt', 'Volantynys', 'day']
chaucer = ''
for i in range(len(L) // 3):
    chaucer = chaucer + L[2 * i][i]
print(chaucer)
```

(A) `fa`
(B) `fi`
(C) `or`
(D) `fay`
(E) `ftw`
(F) `for`
(G) `ftwosVd`
(H) some other characters are printed
(I) an error occurs inside the loop body

```
s = 'Je suis desja d\'amour tanne'
print(s[s.find('j') : s.find('j') + 6])
```

(A) `Je sui`
(B) `Je suis desja d'a`
(C) `jad'am`
(D) `ja d'`
(E) `ja d'a`
(F) `ja d'am`
(G) some other characters are printed
(H) an error occurs during the assignment statement

```
1. B    2. G    3. A    4. A    5. E
```

Question 2. [4 marks]

Part (a) [2 marks]

You need to write a function that returns the number of characters that two strings have in common. Both strings are parameters to the function. Do Step 1 of the Function Design Recipe: write two example function calls and their expected results. As always, choose a good name for your function.

You do not need to write any other steps of the Function Design Recipe.

```
    """



    >>> count_overlap('abc', 'abd')
    2
    >>> count_overlap('abc', 'def')
    0
    """
```

Part (b) [2 marks]

You've decided you want to write a function that capitalizes the first number of characters in a string, where the string is made up only of alphabetic characters. Below are some example calls such as you might produce during Step 1 of the Function Design Recipe. Fill in the function header, including the type contract. As always, select good parameter names.

You do not need to write a description. You do not need to write the function body.

```
def capitalize_first_letters(s: str, count: int) -> str

    """
    >>> capitalize_first('abc', 2)
    'ABc'
    >>> capitalize_first('aa', 0)
    'aa'
    """

    # DO NOT WRITE THE FUNCTION BODY
```

Question 3. [4 marks]

Complete the following function according to its docstring.

```
def bagel_order(bagel_type: str, cream_cheese: str, toasted: bool) -> str:
    """Return the bagel order with the given bagel_type, cream_cheese, and
    whether or not it is toasted.

    If toasted is False, the format is as follows:
    <bagel_type> bagel with <cream_cheese> cream cheese

    If toasted is True, the format is as follows:
    <bagel_type> bagel toasted with <cream_cheese> cream cheese

    If cream_cheese is '', then use 'regular'.
    If cream_cheese is 'no', then omit the last part of the string.

    There should exactly one space between each word in the order, and no
    extra leading or trailing spaces.

    >>> bagel_order('poppy seed', '', True)
    'poppy seed bagel toasted with regular cream cheese'
    >>> bagel_order('everything', 'light', False)
    'everything bagel with light cream cheese'
    >>> bagel_order('plain', 'no', True)
    'plain bagel toasted'
    """

    s = bagel_type + ' bagel'
    if toasted:
        s = s + ' toasted'
    if cream_cheese != 'no':
        if cream_cheese == '':
            cream_cheese = 'regular'
        s = s + ' with ' + cream_cheese + ' cream cheese'
    return s
```

Question 4. [3 marks]

Fill in the box with the while loop condition required for the function to work as described in its docstring.

```
def find_digit(word: str) -> int:
    """Return the index of the first digit character in word, or the length of word
    if it does not contain any digit characters.

    >>> find_digit('!Ba4262')
    3
    >>> find_digit('123Hello')
    0
    >>> find_digit('cats!')
    5
    """
    i = 0
    while                                                                          :
        i = i + 1
    return i



    i = 0
    while i < len(word) and not word[i].isdigit():
        i = i + 1
    return i
```

Question 5. [5 marks]

Complete the function body below according to its docstring. Hint: consider using `range` on your answer.

```
def has_pair(s: str) -> bool:
    """Return True if and only if s has 2 consecutive characters
    (i.e., next to each other) that are the same, and False otherwise.

    >>> has_pair('programming!')
    True
    >>> has_pair('Llama')
    False
    """

    for i in range(len(s) - 1):
        if s[i] == s[i+1]:
            return True
    return False
```

Question 6. [3 marks]

Complete the following function according to its docstring, **without using the method** `str.count`.

```
def count_alphanumeric(s: str) -> float:
    """Return the percentage of characters in s that are alphanumeric
    (letters or digits).
    The percentage should be between 0.0 and 1.0.

    Precondition: len(s) >= 1

    >>> count_alphanumeric('csc108')
    1.0
    >>> count_alphanumeric('I love 108')
    0.8
    >>> count_alphanumeric('!!!')
    0.0
    """

    num_alnum = 0
    for ch in s:
        if ch.isalnum():
            num_alnum = num_alnum + 1

    return num_alnum / len(s)
```