Question 1. [10 MARKS]

For the purpose of this question, we will consider the letters"a", "e", "i", "o" and "u" (whether lowercase or uppercase) to be vowels, but not "y".

Part (a) [4 MARKS]

Consider the following function:

```
def vowels(s, i):
    '''Return the length of the longest sequence of consecutive vowels within
    s that starts at index i. s is a str of length at least one, and i is a
    valid index into s.'''
```

Complete the table below by adding four distinct test cases for function `vowels`. For each, provide specific values for `s` and `i`, the expected function result, and the purpose of the test case.

All of the test cases should be different from each other, and each should test something significant. You will receive no credit for repetitive test cases. Do not include tests that check for invalid input.

Value of `s`	Value of `i`	Expected result	Purpose of this test case

There are many more than 4 good test cases. Attributes of the inputs that can be varied:

- length of s: 1, 2, larger.
- value of i: 0, len(s), in between
- length of run: 0, 1, longer
- case: run involves all lowercase, or mixed upper and lower case

Part (b) [6 MARKS]

Write the function `vowels` according to its docstring. You must not use a for loop, and must not use `break`. Answers that do will receive no credit.

```
def vowels(s, i):
    '''Return the length of the longest sequence of consecutive vowels within
    s that starts at index i. s is a str of length at least one, and i is a
    valid index into s.'''

    lowercase_s = s.lower()
    j = i
    while j < len(s) and lowercase_s[j] in 'aeiou':
        j = j + 1
    return j - i
```

Question 2. [10 MARKS]

Part (a) [6 MARKS]

Write the following function according to its docstring:

```
def process_students(r):
    '''r is an open reader with data about students: their name, cdf account,
    age, college and home city. Each line has the following format:

    name,cdf,age,college,city

    There are no commas other than the ones used as separators.

    Return a dictionary in which each key is a college and its value
    is the list of cdf accounts for students at that college.'''

    d = {}
    for line in r:
        items = line.split(",")
        cdf, college = items[1], items[3]
        if college in d:
            d[college].append(cdf)
        else:
            d[college] = [cdf]
    return d
```

Part (b) [4 MARKS]

Write a program that opens a file called "students.txt" that is in the format described above, calls your function to build the dictionary, and pickles the dictionary to a file called "students.pck". You may assume that `cPickle` has been imported and that function `process_students` has been defined.

```
if __name__ == "__main__":
    f = open("students.txt", "r")
    d = process_students(f)
    pickle_file = open("students.pck", "w")
    cPickle.dump(d, pickle_file)
    pickle_file.close()
```