1. Short answer. 2 points each. No partial credit

A. -5
B. 6 2.5
C. 9 9
D. 8
E. 2 0 8
F. 5
G. False
H. True False
I. 1.5
J. Runtime error (cannot convert 'Three' to an int)
K. 3.5 10.0 5.5 (Must be 10.0 not 10)
L. 4 32
M. A ('Wed' is evaluated as True in a Boolean context.)
N. 5
O. 21
P. -1 2
Q. 16m
R. 6 then a runtime error occurs
S. 4 6 2
T. 18 2
U. 5 2
V. variable zz is not defined in the client code (or words to that effect)
W. 11
X. 13
Y. A, B, C (all of them)

Grading Acronyms:

NN -> Not Necessary. Meaning the code performs unnecessary computations or code was written that was not required by the question.

OBOE -> Off By One Error. Very common logic error. The code does one more or one fewer computation than desired. Or accessing an index that is one more or one less than desired.

MCE -> Major Conceptual Error. Answer is way off base, code answers a question other than the question posed, or significant misunderstanding of concept or question.

2. calculate GPA: (defining a function, simple operations, returning a value from a function)

```
# Question 2, get gpa.
def get_gpa(a, b, c, d, f):
    # All classes are 3 credits so that factors out.
    return ((a * 4 + b * 3 + c * 2 + d)
            / (a + b + c + d + f))
```

Criteria:

- header correct, 2 points (lose if ask user for values)
- Calculation of grade points earned correct. 4 points
  - -1 if loop for repeated addition instead of *
  - -1 if multiply number of F's by 0. (unnecessary computation)
  - -1 if multiply number of grades by 3, but not grade points
- Calculation of credit hours correct. (Add all parameters) 3 points
- Does not unnecessarily multiply number of each grade by 3 for credit hours, 1 point
- Use / instead of // for division, 1 point
- Return statement with result, 1 point (lose if print result)
- Using a symbol other than * for multiply

Common problems:
If credit hours *3 need to multiply grade points by 3 as well. An A in a 3-credit class is worth 12 grade points not 4.

Parameters to function have number of As, Bs, Cs, Ds, and Fs. No user input.

Can simply calculate from the sum of the number of the various grades.

No user input. Function shall have parameters for the number of each letter grade.

No need for main or user input. Please just complete the requested function.

3. Wind speed (user input and conditionals)

```
speed = int(input('Enter wind speed: '))
if speed < 0:
    print('Invalid wind speed')
elif speed < 3:
    print('Calm')
elif speed < 10:
    print('Light')
elif speed < 20:
    print('Moderate')
elif speed < 45:
    print('Strong')
else:
    print('Violent')
```

Criteria:
- Ask use for input with input function and prompt, 1 point
- Convert input to int, 1 point
- Store wind speed in a variable, 1 point
- correctly uses if/elif/elif/elif/else structure. Lose if multiple ifs., 3 points
    - -1 if last branch is elif instead of simply else
- Boolean expresisons for classification are correct. Multiple approaches possible. 4 points
    - -1 if unnecessary checks performed, for example
      if speed < 3:
      elif 3 <= speed < 10: # we know speed is >= 3 here, no need to test
- print error message correctly, 1 point
- print other classifications correctly, 1 point
- use of the range function, -3 (not allowed per question)

Common problems:

Use if/elif/elif/else structure for picking one of multiple options to avoid unnecessary computations.

No main function, necessary, just the code snippet.

wind_speed >= 0 or < 3 is not valid Python syntax. -> wind_speed >= 0 or wind_speed < 3

Misunderstanding o Boolean context: wind_speed < 10 or 5 is **always** true. 5 is seen as true

All non-negative speed would print Calm. Should be and not or.

4. Dice rolling (while loops, cumulative sum):

```
# Question 4, roll a die until the sum of the rolls is >= goal.
# Return the number of rolls it took to reach the goal.
def rolls_to_get_total(sides, goal):
    total = 0
    count = 0
    while total < goal:
        total += randint(1, sides)
        count += 1
    return count
```

Criteria:

- variable for total of rolls, initialized correctly, 1 point
- variable for number of rolls, initialized correctly, 1 point
- while loop with correct Boolean expression, 3 points
    - -1 if off by one error, total <= goal
    - -2 if infinite loop due to total != goal
- increment number of rolls variable in loop, 2 points
- call random.randint(1, sides) correctly, 2 points
- add result of randint function to running total, 2 points
- return number of rolls, 2 points
- output of any kind, -1
- getting user input instead of using given parameters, -2

Common problems:

Did not declare and initialize variable for tracking the number of rolls before the loop

Off By One Error - if we reach goal exactly we can stop. <= can lead to one unnecessary roll.

If we set condition to total != goal an infinite loop can occur if we don't reach goal exactly. Stop total is >= goal. Keep going only if total < goal.

random.randint(1, sides) not 6. Might not be a 6-sided die.

If the die is rolled once outside the loop then count must be set to 1 not 0.

5. Rectangle areas (nested for loops):

```
# Question 5, print out the areas of all squares with integer widths
between
# [1, max_width] and integer heights between [1, max_height].
def print_areas(max_width, max_height):
    for width in range(1, max_width + 1):
        for height in range(1, max_height + 1):
            area = width * height
            print('width =', width, 'height =', height,
                  'area =', area)
```

Criteria:

- outer for loop for width correct. lose if outer loop is height, 3 points
  - -1 if off by one error max_width instead of max_width + 1
- Nested loop. Actually nested and correct. 3 points
  - -1 if off by one error max_height instead of max_height + 1
- calculate area correctly, 2 points
- print out widths, heights, and areas correctly. 2 points
- Correct number of lines printed out, 2 points
- Asking for user input, -2

Common problems:

OBOE Off By One Error, range(1, max_width) instead of range(1, max_width + 1)

Wrong values for range: range(0, max_width) or range(max_width) will have widths from 0 inclusive to max_width exclusive. Output did not show output of width of 0.

str function not allowed

area calculation and call to print must be indented to be controlled by the inner loop.

Must print width for each area calculated, not just for each width. Must print out current width on each line.

```
for width in range(1, max_width + 1):
    print('width =', width, end='') # Does not work as intended.
        for height in range(1, max_height + 1):
```