CSC 110

Midterm Exam:
Thursday, 14 October 2021

Name:_____SOLUTION_____(please print clearly!)

UVic ID number:_____(please print clearly!)

Signature:_____

Exam duration: 50 minutes

Instructor: Celina Berg

Students must check the number of pages in this examination paper before beginning to write, and report any discrepancy immediately.

- We will not answer questions during the exam. If you feel there is an error or ambiguity, write your assumption and answer the question based on that assumption.
- Answer all questions on this exam paper.
- The exam is closed book. No books or notes are permitted.
- **Electronic devices, including calculators, are not permitted**.
- The marks assigned to each part are printed within brackets. Partial marks are available.
- There are eight (8) pages in this document, including this cover page.
- Page 6 is left blank for scratch work. If you write an answer on that page, clearly indicate this for the grader under the corresponding question.
- Clearly indicate only one answer to be graded. Questions with more than one answer will be given a **zero grade**.
- It is strongly recommended that you read the entire exam through from beginning to end before beginning to answer the questions.

Part 1 (15 marks)

For the following questions, write your final answer in the box provided.
Write "invalid" for those with syntax errors.
RUBRIC:
1 mark per box - no part marks
Exception for Question g) as follows:

-0.5 mark for each error in statement to a minimum of 0
0 marks given for calling round - round does not print 4 decimal places
For example, try:

```
x = 4.2
print(f'{x: .4f}') # prints 4.2000 versus using round will print 4.2
```

a) Given the following variable declarations and initializations:

```
a = 1.5
b = 2
c = 5
d = 'boo'
e = 10
```

What does each expression evaluate to?

i. `a + b ** b * c` — 21.5

ii. `c % b < a` — True

iii. `e / c * d` — invalid

iv. `d * (b + c // b)` — 'booboobooboo'

b) What are the values of the variables a, b and c after the following code segment has executed?

```
a = 4
b = 8
c = b/a
c = a
a = b
```

a: 8

b: 8

c: 4

c) What are the values of the variables d and e after the following code segment has executed?

```
d = -4
e = 5

if d < 0:
```

d: 4

e: 0

```
    d *= -1

if d > 3 and e <= 5:
    e -= e
else:
    e += e
```

d) What are the values of the variables f and g after the following code segment has executed?

```
f = 10
g = -2

if f > 0:
    if g > 0:
        g = g * 2
else:
    f = 0

g = g * -1
```

f: 10

g: 2

e) What is the value of the following expression assuming that m has the value 4 and n has the value 9?

```
m > 0 and n < 10 and not(m > n)
```

True

f) Given the following function definition (documentation omitted intentionally):

```
def loop_compute(x, y):
    val = 0

    for num in range(y, x, -x):
        val += num
```

```
    return val
```

What value is returned by the function on each of the following calls?

`loop_compute(1, 1)`	0
`loop_compute(2, 5)`	8

g) Assume the variable distance has been declared and initialized. Write the line of code to print the value of distance to 4 decimal places.

```
print(f'{distance:.4f}')
```

Part 2 (17 marks)

Design a function called `generate_sequence` that takes the following 3 arguments:

- the number of machine readings required
- the starting time of the first machine reading
- the threshold machine reading limit

This function is to return a string containing the specified number of machine readings at the hours beginning at the given starting time and increasing by 1 for each reading.

A machine reading for a given time can be collected by calling the `get_machine_reading` function (documentation below).

Any readings above the given threshold machine reading limit are to be multiplied by a scale factor of 0.8 before adding them to the sequence. Any readings at or below the given threshold machine reading limit are to be multiplied by a scale factor of 1.5 before adding them to the sequence.

The function assumes the times of the machine readings required will be a valid hour in the 24 hour clock (ie. 1 to 24 inclusive).

Example:
If the function is called as: `generate_sequence(5, 2, 32.9)`
it should get the machine readings at times: 2, 3, 4, 5 and 6 and add those readings to a result string, multiplying any readings above 32.9 by the scale factor of 0.8
and multiplying any readings at or below 32.9 by the scale factor of 1.5
and the final result string is returned.

For example, if the following 5 machine readings collected were: 3.4, 8.23, 33.4, 5.8, 40.0
the function would return the string: `'5.1, 12.345, 26.52, 8.7, 32.0'`

```
def get_machine_reading(time: int) -> float:
    """
    returns the machine reading at the given time where time is an hour
    on a 24 hour clock
    Precondition: 1<=time<=24
    """
    # implementation and examples omitted intentionally
    # DO NOT attempt to finish this function
```

Complete your function design in the box provided on the following page **including required documentation.**
Due to the ambiguity of the values to returned by the `get_machine_reading` function,
you do not need to provide example tests within your documentation.
Assume the implementation of the `get_machine_reading` function is within the same file.

Model solution – other valid solutions:

```
SCALE_LOWER  = 0.8
SCALE_HIGHER = 1.5

def generate_sequence(num_readings: int, start_time: int, limit: float) -> str:
    """ returns a sequece of num_readings machine readings separate by commas.
    If any of the machine readings are above the given limit,
    the reading is scaled by SCALE_LOWER.
    If any of the machine readings are at or below the given limit,
    the reading is scaled by SCALE_HIGHER.
    Precondition: 1 <= start_time <= 24, start_time+(n-1) <= 24
    Examples omitted intentionally due to ambiguity of get_machine_reading
    """
    sequence = ''
    for count in range(num_readings):
        if count > 0:
            sequence += ', '
        reading = get_machine_reading(count + start_time)
        if reading > limit:
            reading *= SCALE_LOWER
        else:
            reading *= SCALE_HIGHER

        sequence += str(reading)

    return sequence
```

Rubric:

/3 documentation
- 1 mark function header including type hints
- -0.5 per error to minimum of 0
- 1 mark docstring containing purpose statement
- 1 mark CONSTANTS to replace scale factors magic numbers
- no marks deducted if declared as local variable instead

/2 initializes and returns a result string
- 1 mark - initialization of the accumulator to an empty string
- 1 mark - return the accumulated string

/3 loop repeats correct number of times
- 1 off by one in number of iterations (1 too many or 1 too few)
- 1 if special case above the loop (does not work for 0 number of readings)
- 2 if loop limit wrong ie. calls range with (start time, number of readings) - not enough repetitions
- 1 for return statement inside the loop
- 1 using 'while in range'...
- 2 if missing call to range

/2 marks - conditionally adds comma to result string
- -1 if trailing comma or incorrect condition for adding comma

/2 marks - calls get_machine_reading function
- -1 if called multiple times instead of storing result for use
- -1 if attempts to use returned value incorrectly
- -1 if error in arguments passed

/2 marks - adds machine reading to result string
- -1 if does not convert number to string
- -1 if string returned when adding to it

/3 marks - conditionally updates machine readings above limit
- -1 if error in relational operator (ie. < instead of <=, < instead of >)
- -1 math error
- -1 if redundant conditional check
- -1 if missing variable update (ie. = instead of *= or no assignment at all)
- -2 comparing wrong variables

Page left blank intentionally for scratch work if needed…

Part 3 (5 marks)

Recall the formula for the distance between two points (x_1, y_1) and (x_2, y_2) is:

$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

The function `get_distance` below should return the calculated distance between two points and should return a -1 if any of the x, y coordinates are negative.

Although it passes all of the tests given within the docstring, these tests do not provide full test coverage and therefore the function body contains logic errors (code errors that cause the function to generate the wrong result according to the specification).

To answer this question:

- Find and fix the logic errors by crossing out incorrect code and writing the code that would replace it.
- In the blank space provided, write the additional tests in `doctest` expected format to ensure **full** test coverage and therefore would have identified these errors.

```
def get_distance(point1_x: int, point1_y: int,
                 point2_x: int, point2_y: int) -> float:
    """
    returns -1 if any given point coordinate is negative, otherwise
    returns the distance between the point at coordinates (point1_x and point1_y)
    and another point at coordinates (point2_x and point2_y)
    >>> get_distance(-1, -4, -5, -2)
    -1
    >>> get_distance(0, 0, 0, 0)
    0.0
    >>> get_distance(4, 2, 6, 9)
    7.280109889280518

    NEED a test with each argument being negative where others are 0 or more
    to expose all errors
    >>> get_distance(-1, 4, 5, 2)
    -1
    >>> get_distance(1, -4, 5, 2)
    -1
    >>> get_distance(1, 4, -5, 2)
    -1
    >>> get_distance(1, 4, 5, -2)
    -1
    """
    if point1_x < 0 and point1_y < 0 and point1_x < 0 and point1_y < 0:
    if point1_x < 0 or point1_y < 0 or point2_x < 0 or point2_y < 0:
        return -1

    xdist = point2_x - point1_x
    ydist = point2_y - point1_y

    result = math.sqrt(xdist**2 + ydist**2)

    return result
RUBRIC
/2 marks - fixes errors
   1 mark - or operators changed to and operators
   1 mark - fix variable names
/3 marks - demonstrates test coverage
   -2 if does not have 4 tests, each having exactly one of the arguments <0
   -2 if the 4 tests each have one of the arguments as 0 (wrong boundary)
   -1 if missing only one of the r test having one of the arguments <0
-1 if fixes something that is not an error (ie. adding unnecessary else statement
```

END OF EXAM

For grading purposes, do not fill in:

Part	Value	Mark
1	15	
2	17	
3	5	
Total	**37**	