(1-3) An approximation for the square root of n can be generated using the following equation:

\[ x_{k+1} = \frac{1}{2} \left( x_k + \frac{n}{x_k} \right), \text{where } x_0 = 1 \]

Each value of x should be a better approximation for the square root of n.

Given function `approx_sqrt`:

```python
def approx_sqrt(num, iterations):
    '''TYPE CONTRACT GOES HERE
    Generates an approximate square root of num, a positive integer, via an iterative process that runs iterations times. The approximate square root is returned.
    >>> approx_sqrt(1, 1)
    1.0
    >>> approx_sqrt(4, 1)
    2.5
    >>> approx_sqrt(4, 5)
    2.000000000000002
    '''
    value = num - 1
    for ctr in range(iterations - 2):
        value = .5 * (value + num/value)
    return value
```

(1) supply the type contract that is consistent with the equation:

a) (int, int) -> float   b) (float, float) -> int

c) (int, int) -> None   d) (str, int) -> float

(2) Replace ??-1 with the code needed to implement the approximation.

a) .5  b) iterations  c) num  d) 1

(3) Replace ??-2 with the code needed to implement the approximation.

a) .5  b) iterations  c) num  d) 1

(4) The binary representation of decimal 27 is

a) 11011  a) 11010  a) 1010  a) 1101
(5) What is the result of executing the following Python code:

```python
>>> isinstance(101, float)
```

a) 101  b) int  c) True  d) False  e) update __main__

(6-8) Given the following Python code:

```python
>>> x = 'CIS 210'
>>> id(x)
4391509160
>>> y = x
>>> id(y)
??-1
>>> x = 'the end'
>>> id(x)
??-2
>>> y
??-3
```

(6) 4391509160 refers to a(n)

a) assignment statement  b) function  c) None type  d) keyword  e) memory location

(7) The value printed at ??-1 will also be 4391509160 (yes or no); the value printed at ??-2 will also be 4391509160 (yes or no).

a) yes/yes  b) no/no  c) yes/no  d) no/yes

(8) The value printed at ??-3 will be

a) 4391509160  b) 'CIS 210'  c) 'the end'  d) None

(9-10)

(9) What will be printed when the following Python code is executed?

```python
n = 5
mysum = 0
for ctr in range(1, n):
    mysum += ctr
print(mysum)
```

a) 0  b) 5  c) 10  d) 15  e) None

(10) This code is an example of

a) accumulator pattern  b) TypeError  c) conditional  d) indefinite iteration  e) infinite loop
(11) Given the following Python code:

1 - >>> ftemp = 212
2 - >>> ctemp = (ftemp - 32) * 5/9
3 - >>> ctemp = ftemp - 32 * 5/9

The value of ctemp will [??] from line 2 to line 3; the type of ctemp will [??] from line 2 to line 3

a) stay the same/change       b) change/stay the same       c) stay the same/stay the same       d) change/change

(12-13) Given the following Python code:

0   import math
1
2   def isInCircle(x, y):
3       '''(number, number) -> ??
4
5       Quiz.
6
7
8   d = math.sqrt(x**2 + y**2)
9   isIn = (d <= 1)
10   return isIn

(12) Complete the type contract:

a) int  b) float  c) number  d) bool  e) str

(13) Indicate which lines of code would need to be changed for isInCircle to check whether point (x, y) were inside a circle with a radius of any length.

a) 0, 10  b) 2, 3, 9  c) 2, 9, 10  d) 2, 9  e) 8, 9

(14-15) Given the following Python code:

    def twice(x):
        ''' quiz '''
        result = 2 * x
        return result

>>> x = 99
>>> twice(10)  a) 99  b) 10  c) 20  d) 198  e) NameError
??-1

>>> x  a) 99  b) 10  c) 20  d) 198  e) NameError
??-2

(14) Replace ??-1 with the expected result:

(15) Replace ??-2 with the expected result:
(16) Given the following Python code:

```python
def q30(score):
    ''' quiz '''

    gradepoint = 0

    if score >= 90:
        gradepoint = 4
    elif score >= 80:
        gradepoint = 3
    elif score >= 70:
        gradepoint = 2
    elif score >= 60:
        gradepoint = 1

    return gradepoint
```

What is the result of executing `>>> q30(80)`?

a) 4  b) 3  c) 2  d) 1  e) NameError