THE UNIVERSITY OF QUEENSLAND
AUSTRALIA

This exam paper must not be removed from the venue

Venue _____

Seat Number _____

Student Number |__|__|__|__|__|__|__|__|

Family Name _____

First Name _____

School of Information Technology and Electrical Engineering

EXAMINATION

Semester Two Final Examinations, 2018

CSSE1001 / CSSE7030 Introduction to Software Engineering

This paper is for St Lucia Campus students.

Examination Duration: 90 minutes

Reading Time: 10 minutes

Exam Conditions:

This is a Central Examination

This is a Closed Book Examination - no materials permitted

During reading time - write only on the rough paper provided

This examination paper will be released to the Library

Materials Permitted In The Exam Venue:

(No electronic aids are permitted e.g. laptops, phones)

Calculators - No calculators permitted

Materials To Be Supplied To Students:

1 x Multiple Choice Answer Sheet

Instructions To Students:

Additional exam materials (eg. answer booklets, rough paper) will be provided upon request.

For Examiner Use Only

Question	Mark

Total _____

For all questions, please choose the most appropriate answer if it appears that more than one option is a potentially correct answer. All coding questions relate to the Python 3 programming language. If an evaluation produces an error of any kind, choose Error as your answer. Different questions may have different numbers of choices. Each question is worth one mark.

1. What does the expression `(7.1 + 3.3) // 4` evaluate to?

a) `2`
b) `2.0`
c) `2.6`
d) `Error`

2. What does the expression `8 - 3 / 2` evaluate to?

a) `6.5`
b) `6`
c) `2.5`
d) `2`
e) `Error`

3. What does the expression `3**2 % 4` evaluate to?

a) `0`
b) `1`
c) `2`
d) `2.25`
e) `Error`

4. What does the expression `4*'fox'/4` evaluate to?

a) `'fox'`
b) `'foxfoxfoxfox'`
c) `'foxfoxfoxfox/4'`
d) `Error`

5. What does the expression `list(('a', 'c') + (1,2,))` evaluate to?

a) `('a', 'c', 1, 2)`
b) `('a', 'c', 1, 2,)`
c) `['a', 'c', 1, 2]`
d) `['a', 'c', 1, 2,]`
e) `Error`

6. What is the result of `1 < 2 and not 5 != 4` ?

a) `0`

b) `True`

c) `False`

d) `Error`

7. What is the value of `y` after the following statements are evaluated?

```
x = [1, 2, 3, [4, 5, 6]]
y = x[3, 2]
```

a) `5`

b) `6`

c) `3`

d) `[4, 5]`

e) `Error`

8. What is the value of `y` after the following statements are evaluated?

```
v = (['a', 'b'],[ 'c', 'd'])
x,y = v
z=y
z[-1] = 3
```

a) `0`

b) `['c',3]`

c) `['a', 'b']`

d) `['a', 'b', 'c', 3]`

e) `Error`

9. After the assignment `x = 'Just in time'`, which of the following statements assigns `'st'` to `y`?

a) `y = x[3:5]`

b) `y = x[2:4]`

c) `y = x[-7:-9]`

d) `y = x[-10:-8]`

e) `More than one of the above is correct.`

10. What does the following expression evaluate to?

```
('abcdef' + 'ghi')[2 : 6 : 2]
```

a) `'bd'`

b) `'ce'`

c) `'bdf'`

d) `'ceg'`

e) `None of the above`

11. Given the following code:

```
x = input("Please enter a number: ")
y = input("Please enter a second number: ")
print ("x / y =", x / y)
```

and assuming that the user inputs 10 and 4 respectively, what would be the output?

a) `x / y = 2`

b) `x / y = 2.5`

c) `x / y = 0.5`

d) `Error`

12. What is the value of d after the following statements are evaluated?

```
d = {1:'a', 2:'b', 3:'c'}
d[4] = 'd'
d.get(5, 'e')
```

a) `{5:'e'}`

b) `{1:'a', 2:'b', 3:'c'}`

c) `{1:'a', 2:'b', 3:'c', 4:'d'}`

d) `{1:'a', 2:'b', 3:'c', 4:'d', 5:'e'}`

e) `Error`

13. After the assignment `z='one \n life'` what does the expression `sorted(z.split())` evaluate to?

a) `['\n', 'life', 'one']`

b) `['life', 'one', '\n']`

c) `['life', 'one']`

d) `'\n', 'life', 'one'`

e) `Error`

14. What is the value of y after the following code is executed?

```
def g(x):
    a = 11
    x += [2]
    return a

a = 10
b = [3, 4]
g(b)
y = b + [a]
```

a) [3, 4, 2, 10]

b) [3, 4, 2, 11]

c) [3, 4, 10]

d) [3, 4, 11]

e) None of the above

15. What is the value of x after the following code is executed?

```
def f(v, u, b) :
    v.pop(u)
    v = v + [b]
    return v

x = [2, 3, 5]
x = f(x, 2, 3) + x
```

a) [2, 3, 3, 2, 3, 5]

b) [2, 3, 5, 2, 3]

c) [2, 3, 5, 3, 2]

d) [2, 3, 3, 2, 3]

e) Error

16. What is the value of x after the following is evaluated?

```
d = {'Jack Brown' : {'Student No': 532762, 'Courses':
['CSSE1001', 'ECON1010']}, 'Edith Slin': {'Student No':
743672, 'Courses': ['CSSE1001', 'SCIE1000']}}
x=d.get('Jack Brown').get('Courses',[])
```

a) []

b) ['CSSE1001', 'ECON1010']

c) None

d) {}

e) Error

17. What is the value of x after the following is evaluated?

```
d = {'Jack Brown' : {'Student No': 532762, 'Courses':
['CSSE1001', 'ECON1010']}, 'Edith Slin': {'Student No':
743672, 'Courses': ['CSSE1001', 'SCIE1000']}}
x=d.get('Matthew Gown',{}).get('Courses',[])
```

a) ['CSSE1001', 'ECON1010']

b) []

c) None

d) {}

e) Error

18. What is the value of x after the following is evaluated?

```
d = {'Jack Brown' : {'Student No': 532762, 'Courses':
['CSSE1001', 'ECON1010']}, 'Edith Slin': {'Student No': 743672,
'Courses': ['CSSE1001', 'SCIE1000']}}
x=d.get('Jack Brown',{}).get('Courses',[]).append('ENGG1400')
```

a)['CSSE1001', 'ECON1010']

b)['CSSE1001', 'ECON1010', 'ENGG1400']

c)None

d)Error

19. Which of the following descriptions best describe the purpose of this function?

```
def func() :
    t = 0
    r = int(input('Please input an integer: '))
    while r != 0 :
        if r % 2 == 1 :
            t += r
        r = int(input('Please input an integer: '))
    return t
```

a) It does not do anything as the body of the while loop never executes.

b) It is an infinite loop as the while loop condition can never be false.

c) It returns the sum of all integers entered.

d) It returns the sum of all odd integers entered.

e) It returns the sum of all even integers entered.

The following recursive function definition is used in the next two questions.

```
def rec(x) :
    if len(x) == x[0] :
        return x
    return rec(x[2:] + [x[0]])
```

20. What will the function call `rec([1, 2, 2, 1, 4])` return?

a) `[5, 3, 3, 2, 1]`

b) `[5, 3, 1]`

c) `[2, 4]`

d) `RecursionError` will be raised due to maximum recursion depth being exceeded.

e) None of the above

21. What will the function call `rec([4, 2, 3, 8, 6])` return?

a) `[7, 4, 1, 3, 2]`

b) `[7, 1, 2]`

c) `[3, 7]`

d) `RecursionError` will be raised due to maximum recursion depth being exceeded.

The next 4 questions refer to the following definition.

```
def function1(xs,n,m,p):
    t = []
    s = p/abs(p)
    # s is the sign of p (1 or -1)
    while s*(m-n) > 0:
        t.extend(xs[n])
        n += p
    return t
```

22. What is the value of x after the following is evaluated?

```
x = function1(['d', 'e', 'f', 'g', 'h'], 2, 4, 1)
```

a) []

b) ['f', 'g']

c) ['f', ['g', 'h']]

d) ['f', 'g', ['h', 'd']]

e) Error

23. What is the value of x after the following is evaluated?

```
x = function1(['d', 'e', 'f', 'g', 'h'], 2, 6, 1)
```

a) []

b) ['f', 'g', ['h', 'd']]

c) ['f', 'g', 'h', 'd']

d) ['f', 'g', 'h', 'd', 'e']

e) Error

24. What is the value of x after the following is evaluated?

```
x = function1(['d', 'e', 'f', 'g', 'h'], -1, -4, 2)
```

a) []

b) ['g', 'h']

c) ['g', ['h']]

d) ['h', 'f']

e) Error

25. What is the value of x after the following is evaluated?

```
x = function1(['d', 'e', 'f', 'g', 'h'], 3, 3, 4)
```

a) []

b) ['e', 'g']

c) ['e', 'g', 'f']

d) ['e', 'f', 'g', 'd']

e) Error

The next three questions refer to the following partial definition of a BankAccount class.

```
class BankAccount(object):
    def __init__(self, customer):
        """customer is the name of the account owner """
        self._customer = customer
        # the balance on the account in dollars
        self._balance = 0.0

    def deposit(self, amount):
        """Deposit amount into the account.
        deposit(float) -> None """
        ## line 1 ##

    def get_balance(self):
        """Get the balance.
        get_balance() -> float """
        ## line 2 ##
```

Assume that the following has been evaluated.

```
ed_gal = BankAccount('Edward Gal')
```

26. What is the required code for ## line 1 ## to meet the description in the comments?

a) `_balance += amount`

b) `balance += amount`

c) `self._balance += amount`

d) `balance = balance + amount`

e) `More than one of the above is correct`

27. What is the required code for ## line 2 ## to meet the description in the comments?

a) `print self.balance`

b) `print self._balance`

c) `return self.amount`

d) `return self._balance`

e) `More than one of the above is correct.`

28. Which of the following correctly updates the balance by $35.00 for the object `ed_gal`?

a) `ed_gal.deposit(35.00)`

b) `ed_gal.deposit() + 35.00`

c) `deposit(ed_gal, 35.00)`

d) `deposit(ed_gal) + 35.00`

e) `More than one of the above is correct.`

The next 3 questions refer to the following definition that is missing three lines of code. This function extracts URL information from the body of HTML text as in the example below.

```
>>> text = """
<body>
 <ul>
<li><a href="/future-students">Future Students</a></li>
 <li><a href="/current-students">Current Students</a></li>
</ul>
 </body>
"""
>>> get_urls(text)
 {'Future Students': '/future-students', 'Current Students':
'/current-students'}
>>>
```

The definition of the function `get_urls` is given below.

```
def get_urls(text):
    """Return a dictionary that associates each URL with the text
for

    that URL
    get_urls(str) -> dict(str:str)
    Precondition: text is valid HTML """
    d = {}
    pos = 0
    next_pos = text.find("<a href")
    while next_pos != -1:
        ## line 1 ##
        end_tag = text.find('>', next_pos)
        start_tag = text.find('<', end_tag)
        ## line 2 ##
        d[name] = url
        ## line 3 ##
    return d
```

29. What is the required code for ## line 1 ##?

a) `url = text[next_pos:].split('"', 2)[0]`

b) `url = text[next_pos:].split('"', 2)[1]`

c) `url = text[:next_pos].split('"', 2)[0]`

d) `url = text[:next_pos].split('"', 2)[1]`

e) None of the above

30. What is the required code for ## line 2 ##?

a) `name = text[end_tag:start_tag]`

b) `name = text[end_tag+1:start_tag]`

c) `name = text[end_tag:start_tag-1]`

d) `name = text[end_tag+1:start_tag-1]`

e) `None of the above`

31. What is the required code for ## line 3 ##?

a) `break`

b) `end_tag = text.find('>', start_tag)`

c) `next_pos = text.find("<a href", start_tag)`

d) `end_tag = text.find('>', next_pos)`

e) `next_pos = text.find("<a href", next_pos)`

The next five questions refer to the following class definitions.

```
class A(object) :
    def __init__(self, x, y) :
        self._x = x
        self._y = y

    def a1(self, x, y) :
        return self.a2(x, y) +3

    def a2(self, x, y) :
        return y - x


class B(A) :
    def a2(self, x, y) :
        return x + 2* y


class C(B) :
    def __init__(self, x, y) :
        super().__init__(x, y)
        self._x = x
        self._y = 2*y+x

    def a1(self, x, y) :
        return self._x + self._y


class D(B) :
    def __init__(self, x, y) :
        super().__init__(x, y)
        self._x = x
        self._y = y-1

    def a1(self, x, y)  :
        return x + y +1

    def a2(self, x, y) :
        return super().a2(x, y)

a = A(1,3)
b = B(1, 2)
c = C(2, 1)
d = D(2, 2)
```

32. What does `a.a2(2, 1)` return?

a) `-1`

b) `0`

c) `2`

d) `5`

33. What does `b.a1(1, 1)` return?

a) `2`

b) `4`

c) `6`

d) `8`

34. What does `c.a1(2,2)` return?

a) `4`

b) `5`

c) `6`

d) `7`

35. What does `d.a1(1, 1)` return?

a) `2`

b) `3`

c) `4`

d) `5`

36. What does `d.a2(2, 1)` return?

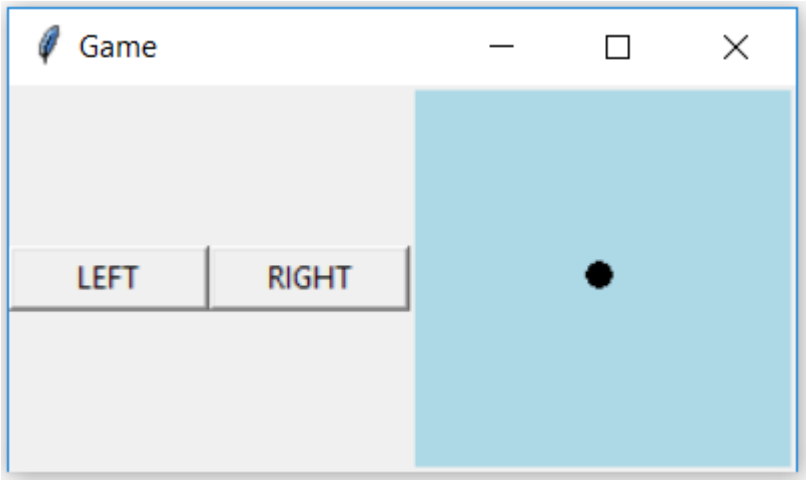
a) `0`

b) `2`

c) `4`

d) `6`

The next two questions relate to the following partial implementation of a GUI. The application has two buttons in a frame and custom canvas on which the circle can be moved left or right by clicking on the buttons. The completed GUI is shown in the image below. The code is provided on the next page.



```
import tkinter as tk

class Screen(tk.Canvas):
    def __init__(self, parent):
        super().__init__(parent, bg="light blue",
                         width=150, height=150)
        self._x, self._y = (width / 2, height / 2)
        self._redraw()

    def _redraw(self):
        """Redraw the screen after a move."""
        self.delete(tk.ALL)
        ## code block 1 ##

    def _move(self, dx, dy):
        """Move the circle by a given amount."""
        self._x += dx
        self._y += dy
        self._redraw()

    def move_left(self):
        self._move(-5, 0)

    def move_right(self):
        self._move(5, 0)


class Controls(tk.Frame):
    BUTTON_WIDTH = 10

    def __init__(self, parent, left, right):
        """Parameters:
            parent (Tk): Window for widget.
            left (method): Callback for "left button".
            right (method): Callback for "right button".
        """
        super().__init__(parent)
        ## code block 2 ##


class GameApp(object):
    def __init__(self, master):
        master.title("Game")
        screen = Screen(master)
        controls = Controls(master, screen.move_left,
                            screen.move_right)
        controls.pack(side=tk.LEFT)
        screen.pack(side=tk.LEFT, expand=True,
                    fill=tk.BOTH)
```

37. What is the required code for `## code block 1 ##`?

a)
```
coords = (self._x - 5, self._y - 5,
          self._x + 5, self._y + 5)
create_oval(coords, fill="black", width=0)
```

b)
```
coords = (self._x - 5, self._y - 5,
          self._x + 5, self._y + 5)
self.create_oval(coords, fill="black", width=0)
```

c)
```
coords = (self._x - 5, self._y - 5,
          self._x + 5, self._y + 5)
Canvas.create_oval(coords, fill="black", width=0)
```

d)
```
coords = (self._x - 5, self._y - 5,
          self._x + 5, self._y + 5)
super().create_oval(coords, fill="black", width=0)
```

e) More than one of the above is correct.

38. What is the required code for `## code block 2 ##`?

a)
```
left_btn = tk.Button(self, text="LEFT",
                     width=10, command=left())
left_btn.pack(side=tk.LEFT)
right_btn = tk.Button(self, text="RIGHT",
                      width=10, command=right())
right_btn.pack(side=tk.LEFT)
```

b)
```
left_btn = tk.Button(self, text="LEFT",
                     width=10, command=Screen.left())
left_btn.pack(side=tk.LEFT)
right_btn = tk.Button(self, text="RIGHT",
                      width=10, command=Screen.right())
right_btn.pack(side=tk.LEFT)
```

c)
```
left_btn = tk.Button(self, text="LEFT",
                     width=10, command=self.left())
left_btn.pack(side=tk.LEFT)
right_btn = tk.Button(self, text="RIGHT",
                      width=10, command=self.right())
right_btn.pack(side=tk.LEFT)
```

d)
```
left_btn = tk.Button(self, text="LEFT",
                     width=10, command=left)
left_btn.pack(side=tk.LEFT)
right_btn = tk.Button(self, text="RIGHT",
                      width=10, command=right)
right_btn.pack(side=tk.LEFT)
```

e) More than one of the above is correct.

This question relates to the incomplete recursive function definition below that provides indexing into a nested list. The following gives examples of calling the function.

```
>>> nested = [[[1, 2], 3], 7, [4, [5, 6]], [8, 9, 10]]

>>> recursiveIndex(nested, [])

[[[1, 2], 3], 7, [4, [5, 6]], [8, 9, 10]]

>>> recursiveIndex(nested, [2,1,0])

5
```

The first example above shows what happens when the second argument (the indexing list) is empty. The second example indexes the 0'th element of the 1'th element of the 2'th element of nested.

39. What expression needs to replace #### in order to complete the following recursive definition for indexing a nested list

```
def recursiveIndex(nested, indexes):
    """Return the element of the required sublist of the
nested list
    (nested) at the position specified by indexes
    Precondition: indexes specifies a "valid" element of the
nested list """
    if indexes == []:
        return nested
    else:
        return ####
```

a) `recursiveIndex(nested[indexes[0]], indexes[:-1])`

b) `recursiveIndex(nested[indexes[0]], indexes[1:])`

c) `recursiveIndex(nested[indexes[-1]], indexes[:-1])`

d) `recursiveIndex(nested[indexes[-1]], indexes[1:])`

e) None of the above

40. What is the value of y after the following has been evaluated?

```
g = lambda x,y: (x+y)
f = lambda x,y: x > y
xs = [0,1,2,3]
y = [g(x,y) for x in xs for y in xs if f(x,y)]
```

a) [1, 1, 2, 2, 3, 3]

b) [1, 2, 4, 5, 3, 1]

c) [1, 2, 3, 3, 2, 1]

d) [1, 2, 3, 3, 4, 5]

e) Error

END OF EXAMINATION