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 Electrical Engineering & Computer Science
Semester One Examinations, 2024
CSSE1001 / CSSE7030 Introduction to Software Engineering

This paper is for St Lucia Campus students.

Examination Duration: 120 minutes

Planning Time: 10 minutes

Exam Conditions:

•No written or printed material permitted
•Casio FX82 series or UQ approved and labelled calculator only
•During Planning Time - Students are encouraged to review and plan responses to the exam questions

Materials Permitted in the Exam Venue:
(No electronic aids are permitted e.g. laptops, phones)

None

Materials to be supplied to Students:
Additional exam materials (e.g. answer booklets, rough paper) will be provided upon request.

1 x Gradescope Bubble Sheet

Instructions to Students:
If you believe there is missing or incorrect information impacting your ability to answer any question, please state this when writing your answer.

Indicate your answer to the first 30 questions on the GradeScope bubble sheet.

For Examiner Use Only

Question	Mark

Total _________

`Error` is the correct answer for any question with code that throws an error of any kind.

Multiple Choice

Question 1. [1 MARK]

What is stored in x after *only* the following is entered into Python?

```
x = (7, 3, (6,)) + (9, (5))
```

A. `(7, 3, 6, 9, 5)`

B. `(7, 3, (6,), 9, (5))`

C. `(7, 3, (6,), (9, (5)))`

[✓ D.] `(7, 3, (6,) , 9, 5)`

E. `Error`

Question 2. [1 MARK]

The following is a recursive function with a partially implemented base case; it computes the sum of a list of numbers. What should we replace `#sub` with to complete this function?

```
def sum(xs: list[int]) -> int:
    """
    >>> sum([1, 2, 3])
    6
    """
    (a, b) = #sub

    if len(xs) == a:
        return b

    return xs[0] + sum(xs[1:])
```

A. `(0, xs[0])`

B. `(1, xs[0])`

C. `(0, 1)`

[✓ D.] `(0, 0)`

E. None of the above.

Question 3. [1 MARK]

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

```
def f(xs: list, x) -> list:
    xs.append(x)
    return xs

x = [5, 9]
x = f(x, 2) + x
```

A. `[5, 9, 2]`

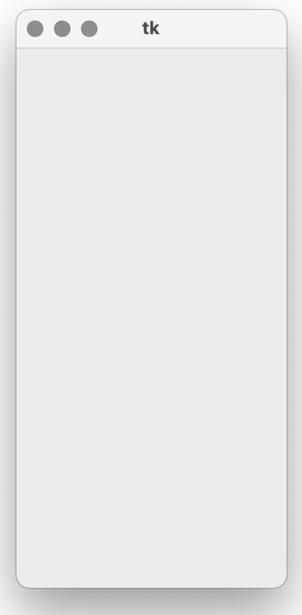
B. `[5, 9, 2, 5, 9]`

[✓ C.] `[5, 9, 2, 5, 9, 2]`

D. `Error`

E. None of the above.

Question 4. [1 MARK]



What line of code should replace `#sub` in order to generate the window illustrated above?

```
import tkinter as tk
root = tk.Tk()
#sub
root.mainloop()
```

[✓ A. `root.geometry("200x400")`]

B. `root.geometry("200 x 400")`

C. `root.geometry("400x200")`

D. `root.geometry("400 x 200")`

E. More than one of the above.

Question 5. [1 MARK]

What is stored in x after *only* the following is entered into Python?

```
d = {'Brown': {'ID': 732, 'Orders': ['chisel', 'spanner']},
     'Black': {'ID': 461, 'Orders': ['lathe', 'crowbar']}}
x = d.get('White').get('Orders')
```

A. `['chisel', 'spanner']`

B. `[]`

C. `['lathe', 'crowbar']`

[✓ D. `Error`]

E. None of the above

Question 6. [1 MARK]

Suppose *only* the following lines of Python have been executed.

```
xs = "champagne problems"
x = #sub
```

What should replace `#sub` so that `'e p'` is assigned to `x`.

A. `xs[8:10]`

B. `xs[8:11]`

C. `xs[-10:-8]`

D. `xs[-10:-7]`

[✓ E. More than one of the above.]

Question 7. [1 MARK]

What is the value of the global variable `a` after the following code is executed?

```
def f(x):
    a = 3
    x = x / a
    return (a+x) % x

a = 9
f(a)
```

A. `0`

B. `0.0`

C. `3`

D. `3.0`

[✓ E. `9`]

Question 8. [1 MARK]

Given the following code:

```
x = input("Prompt: ")
y = input("Prompt: ")
print(f"x - y = {x - y}")
```

If user types `7` then `3`, what is printed?

A. `x - y = 4`

B. `x - y = 7 - 3`

C. `7 - 3 = 4`

D. `x - y = {x - y}`

[✓ E. `Error`]

Question 9. [1 MARK]

What is stored in `y` after *only* the following is entered into Python?

```
x = 'two \t \t pairs'
y = '\t'.join(x.split('\t'))
```

A. `'two \t pairs'`

[✓ B. `'two \t \t pairs'`]

C. `'two\tpairs'`

D. `Error`

E. None of the above.

Question 10. [1 MARK]

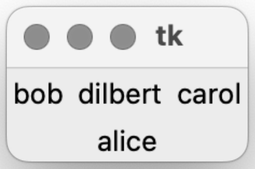
What replaces `#sub` in the following code to generate the image to its right?

```
import tkinter as tk

root = tk.Tk()
(s1, s2, s3, s4) = #sub

tk.Label(text="alice").pack(side=s1)
tk.Label(text="bob").pack(side=s2)
tk.Label(text="carol").pack(side=s3)
tk.Label(text="dilbert").pack(side=s4)

root.mainloop()
```



```
tk
bob dilbert carol
alice
```

A. `(tk.BOTTOM, tk.RIGHT, tk.TOP, tk.BOTTOM)`

[✓ B. `(tk.BOTTOM, tk.LEFT, tk.RIGHT, tk.TOP)`]

C. `(tk.BOTTOM, tk.RIGHT, tk.LEFT, tk.BOTTOM)`

D. `(tk.BOTTOM, tk.LEFT, tk.LEFT, tk.LEFT)`

E. None of the above.

Question 11. [1 MARK]

What is the value of x after the following statements are evaluated?

```
x = 0
for y, z in enumerate([[1], [2]]):
    x += 2 * y + z
```

A. `3`

B. `5`

C. `7`

[✓ D. `Error`]

E. None of the above.

Question 12. [1 MARK]

Suppose `xs` is a list. Which expression evaluates to `True` when `xs` is empty.

[✓ A. `bool(not xs)`]

B. `bool(xs)`

C. `bool(len(xs))`

D. `bool(xs in [])`

E. More than one of the above.

Question 13. [1 MARK]

Suppose we want to define a name for maximum volume that is intended to be private. Which name is most appropriate?

A. `__maximum_volume__`

B. `MaximumVolume`

[✓ C. `_maximum_volume`]

D. `MAXIMUM_VOLUME`

E. `maximumValue`

Question 14. [1 MARK]

What is the value of `y` after *only* the following has been evaluated?

```
z = lambda v, w: v+w
xs = [1,2,3,4]
ys = [3,4,5,6]
y = [z(v,w) for v in xs if v <2 for w in ys]
```

A. `[]`

B. `[4, 6, 8, 10]`

[✓ C. `[4, 5, 6, 7]`]

D. `[4, 5, 6, 7, 5, 6, 7, 8]`

E. None of the above.

Question 15. [1 MARK]

What is the value of `x` after *only* the following code is executed?

```
x = 5.1 + 24.2//6 ** 2
```

A. `5`

[✓ B. `5.1`]

C. `x`

D. `21`

E. `21.1`

Question 16. [1 MARK]

What is the value of x after *only* the following code is executed?

```
x = 1 // 4 * 'drake'
```

[✓ A. `''` (the empty string)]

B. `' '` (a space)

C. `'d'`

D. `'drake'`

E. `Error`

Question 17. [1 MARK]

After starting up the Python interpreter, the following code (and only the following code) is entered.

```
if [] and y:
    y = 0
else:
    y = 1
```

What error, if any, does this code raise?

A. `NameError`

B. `IndexError`

C. `TypeError`

D. `SyntaxError`

[✓ E. This is valid Python code.]

Question 18. [1 MARK]

Consider the following function.

```
def foo(xs: list[int], ys: dict) -> bool:
    """ Precondition:  len(xs) > 0
    """
    for x in xs:
        if not x in ys:
            return True
    return False
```

What best describes the behaviour of `foo` provided it is invoked with all preconditions satisfied?

A. `foo` *always* returns `True`.

B. `foo` *always* returns `False`.

[✓ C. `foo` returns `False` *only* when every element of `xs` is a *key* of `ys`.]

D. `foo` returns `True` *only* when there is an element of `xs` that is a *value* of `ys`.

E. `foo` *always* throws an `Error`.

Question 19. [1 MARK]

Which of the following statements is true?

A. Lists are *mutable* but dictionaries are *immutable*.

B. User defined classes are by default *immutable*.

C. Values and keys in dictionaries must *both* be *immutable*.

D. Strings, integers, floats, booleans and lists are *all* immutable.

E. None of the above.

Question 20. [1 MARK]

What is the value of z after *only* the following code has been executed.

```
xss = ['basket', 'bird', 'balloon']
ys = ['ball']
z = [ys[0] in xs and ys[1] in xs for xs in xss]
```

A. `[True]`

B. `[False]`

C. `[True, False, True]`

D. `[True, True, True]`

E. `Error`

Question 21. [1 MARK]

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

```
x = [0, [1, 2], 3]
y = x[-2, 1]
```

A. `0`

B. `1`

C. `2`

D. `3`

E. `Error`

Question 22. [1 MARK]

Consider the following function.

```
def foo(xs: str) -> None:
    for x in xs:
        with open('file.txt', 'w') as f:
            f.write(x)
    return
```

After calling `foo` without generating an error, which option *can* be the contents of `file.txt`?

A. `aaaa`

B. `wawa`

C. `awwaww`

[✓ D. All of them.]

E. None of the above.

Question 23. [1 MARK]

Consider the docstring, type contract, and usage examples of the following function.

```
def lcs(xs: str, ys: str) -> str:
    """ Return the longest substring
    that both and xs and ys have in
    common.
    >>> lcs("", "potato")
    ''
    >>> lcs("tomato", "potato")
    'ato'
    >>> lcs("ababa", "cbaba")
    'baba'
    """
```

What would you expect `lcs(" ", "eras")` to return?

[✓ A. `""` (empty string)]

B. `" "` (single space)

C. `"eras"`

D. `Error`

E. None of the above.

Question 24. [1 MARK]

What error (if any) will the following code produce when executed by Python?

```
def foo(x: int, xs: list[int]) -> bool:
    return x in xs

foo('', ' ')
```

A. `NameError`

B. `IndexError`

C. `TypeError`

D. `SyntaxError`

[✓ E. This is valid Python code.]

Question 25. [1 MARK]

What is the value of x after *only* the following has been evaluated?

```
x = "goodbye".replace("ood", "ello")
```

Given that

```
replace(self, old, new, count=-1, /)
    Return a copy with all occurrences of substring old replaced by new.

      count
        Maximum number of occurrences to replace.
        -1 (the default value) means replace all occurrences.

    If the optional argument count is given, only the first count occurrences are
    replaced.
```

[✓ A. `"gellobye"`]

B. `"goodbye"`

C. `"hellobye"`

D. `None`

E. `Error`

Question 26. [1 MARK]

What exception should be used at `<Error>` to complete the function?

```
def get_value(dictionary: dict, key: str) -> int:
    """
    Retrieves the value associated with the provided key in the dictionary.
    Continues prompting the user until a valid key is entered.
    """

    try:
        return dictionary[key]
    except <Error>:
        return get_value(dictionary, input("Enter another key: "))
```

A. `NameError`

B. `IndexError`

C. `TypeError`

D. `DictError`

[✓ E. `KeyError`]

Question 27. [1 MARK]

For the following function:

```
def r(x: int, y: int) -> int:
    if x == 0:
        return x * y
    return r(x-5, y) + y
```

What will `r(4, 2)` return?

A. `0`

B. `4`

C. `8`

D. `10`

[✓ E. `RecursionError`]

Question 28. [1 MARK]

What does `z` get assigned assuming `23` is the two digit number entered:

```
x = input(Input two digit number: ")
y = int(x)
z = y[0]
```

A. `2`

B. `23`

C. `None`

[✓ D. `Error`]

E. None of the above.

Question 29. [1 MARK]

What is the value of z after running the following code?

```
xs = ['a', (3,4), {1: 'b'}]
ys = xs.copy()
ys[2] = {2: 'c'}
z = xs[2][1]
```

A. `'a'`

[✓ B. `'b'`]

C. `'c'`

D. `Error`

E. None of the above.

Question 30. [1 MARK]

What is the purpose of "setter" methods as they pertain to objects?

[✓ A. They are used to *change* the value of a private variable.]

B. They are used to *retrieve* the value of a private variable.

C. They allow private variables to be shared among multiple instances of the same class.

D. They are used to read data from files.

E. More than one of the above.

The following will be used to match your exam with your name. Please use BLOCK LETTERS *and write as legibly as possible.*

Student Number

Family Name

Given Name

Fill in the Blank

The next *five* questions refer to the following class definitions.

```
class A(object):
    def __init__(self, x):
        self._x = 2 * x

    def f(self, x):
        return self.g(x) + 2

    def g(self, x):
        return x - 1

class B(A):
    def g(self, y):
        return self._x + y

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

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

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

    def f(self, y) :
        return self._y + y

    def g(self, x):
        return super().g(x) - x

a = A(1)
b = B(2)
c = C(3, 4)
d = D(5, 6)
```

Write a *single number* in the answer box *and nothing else*.

Question 31. [1 MARK]

What does `a.f(4)` return?

Question 32. [1 MARK]

What does `b.g(3)` return?

Question 33. [1 MARK]

What does `c.f(2)` return?

Question 34. [1 MARK]

What does `d.f(1)` return?

Question 35. [1 MARK]

What does `d.g(0)` return?

Full Solution

Question 36. [5 MARKS]

Implement the following function according to its specification.

```
def foo(xs: str, ys: str) -> bool:
    """
    Given two strings xs and ys, return true only when xs is equal
    to ys when typed into an empty text editor interpreting '!' as
    typing a backspace character.

    For example:
        >>> foo("ab!c", "ac")
        True
    because "ab!c" becomes "ac" when typed.

        >>> foo("ab!!", "ab")
        False
    because "ab!!" becomes "" (empty string) when typed:

        >>> foo("a!c", "c")
        True
    because "a!c" becomes "c" when typed.

    Note that backspacing on the empty string produces the empty string.
    """
```

Write your answer on the next page.

Write your answer on the next page.

Write your answer on the next page.

END OF EXAMINATION