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

Semester Two Examinations, 2022

CSSE1001/7030 Introduction to Software Engineering

This paper is for St Lucia Campus students.

Examination Duration: 120 minutes

Planning Time: 10 minutes

Exam Conditions:

•This is a Closed Book examination - no written materials 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
•This examination paper will be released to the Library

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.

Answer all questions on the sheet provided.

Error is the correct answer (provided it is an option) for any question with code that throws an error of any kind.

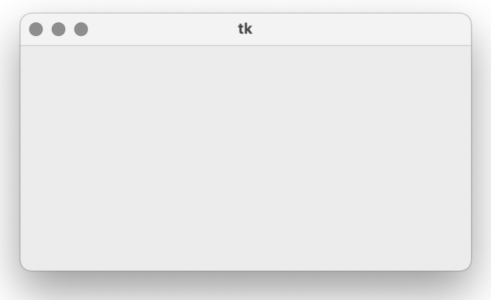
For Examiner Use Only

Question	Mark

Total _________

`Error` is the correct answer (provided it is an option) for any question with code that throws an error of any kind.

Question 1. [1 MARK]



root. geomethy ("widthxhighte)

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 2. [1 MARK]

What is the value of `ans` after running the following code?

```
xs = [3, 5, 7, 9]
ans = 0
for x in xs:
    ans += x // 2
```

3//2=1
1+5//2=3
3+7//2=6
6+9//2=10

[✓ A.] 10

B. 10.5

C. 11

D. 11.5

E. 12

Question 3. [1 MARK]

Consider the following function.

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

What statement best describes the behaviour of `foo`?

A. `foo` always returns `True`.

B. `foo` always returns `False`.

C. `foo` returns `True` when every element of `xs` is a value of `ys`.

[✓ D.] `foo` returns `True` when every element of `xs` is a key of `ys`.

E. None of the above.

Question 4. [1 MARK]

Which option will throw an `IndexError` in the following code when replacing `#sub`?

```
xs = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
#sub
```

max index is 9,

[✓ A.] `xs[len(xs)]`

B. `xs[1-len(xs)]` xs[-9] = 1

C. `xs[len(xs)-1]` xs[9] = 8

[✓ D.] `xs[-1-len(xs)]` -11 out of index

[✓ E.] More than one of the above.

Question 5. [1 MARK]

Supposing Pep8 guidelines have been followed what can be deduced about the name `FooBar`?

`FooBar` is a . . .

A. global variable.

B. constant.

[✓ C.] class name.

D. method.

E. None of the above.

Question 6. [1 MARK]

Suppose the following assignment has been made.

```
xs = "abcdefgh"
```

What list slice of `xs` produces `'hfd'`.

A. `xs[1:7:-2]`

[✓ B.] `xs[7:1:-2]`

C. `xs[6:0:-2]`

D. `xs[0:6:-2]`

E. More than one of the above.

Question 7. [1 MARK]

What value gets assigned to `x`?

```
x = 35 * 2 % 5 ** 2
```

% 除余

/ 除

// 向下取整

A. 0

[✓ B.] 20

C. 70

D. 140

E. None of the above.

Question 8. [1 mark]

How many of the following expressions evaluate to True? Note: Expressions that throw errors do *not* evaluate to True.

```
bool(not [] and "hello")
bool(True or 1/0)
bool(" ")
bool(1 > False)
```

A. 0

B. 1

C. 2

D. 3

E. 4

Question 9. [1 mark]

What is the value of ans after executing the following.

```
ans = 0
for x in range(10):
    if x == 0 or 1 or 9 or 10:
        ans += 1
```

A. 0

B. 3

C. 4

D. 10

E. None of the above.

Question 10. [1 mark]

What is the value of x after executing the following.

```
x = (1, 2, 3) + (4, 5, 6)
```

A. `21`

B. `(5, 7, 9)`

C. `(1, 2, 3, 4, 5, 6)`

D. `Error`

E. None of the above.

Question 11. [1 MARK]

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

```
def product(xs: list[int]) -> int:
    """
    >>> product([1, 2, 3])
    6
    """
    (a, b) = #sub
    if len(xs) == a:
        return b
    return xs[0] * product(xs[1:])
```

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

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

C. `(0, 1)`

D. `(1, 1)`

E. None of the above.

Question 12. [1 MARK]

What type of error is thrown by executing the following code?

```
def foo(x: int, y: int) -> int:
    """
    >>> foo(2, 3)
    6
    """
    return x*y

ans = foo("oi", 3)
```

A. `TypeError`

B. `NameError`

C. `ValueError`

D. `IndexError`

E. This is valid Python code.

Question 13. [1 mark]

What is the value of ans after running the following code.

```
def foo(xs: list) :
    if not xs:
        return [[]]
    else:
        ys = foo(xs[1:])
        x = xs[0]
    return [y + [x] for y in ys] + ys

ans = foo([1, 2, 3])
```

A. `[[], [3], [2], [3, 2], [1], [3, 1], [2, 1], [3, 2, 1]]`

B. `[[3, 2, 1], [2, 1], [3, 1], [1], [3, 2], [2], [3], []]`

C. `[[], [3], [2], [2, 3], [1], [1, 3], [1, 2], [1, 2, 3]]`

D. `[[1, 2, 3], [1, 2], [1, 3], [1], [2, 3], [2], [3], []]`

E. None of the above.

Question 14. [1 mark]

What is the value of xs after the following is evaluated?

```
xs = "abc"
ys = xs
ys[1] = "B"
```

A. `""` (empty string)

B. `"abc"`

C. `"aBc"`

D. `Error`

E. None of the above.

Question 15. [1 mark]

What is the value of zs after executing the following?

```
ys = [1, 2]
zs = [4]
ys.extend([3])
zs.append(ys)
```

A. `[4, [1, 2, 3]]`

B. `[4, 1, 2, 3]`

C. `[4, 3, 1, 2]`

D. `Error`

E. None of the above.

Question 16. [1 MARK]

Consider the function `foo` defined below that computes the area of a circle with integer radius. What is the *type* of its return value?

```
def foo(radius: int):
    """
    Precondition:  radius > 0
    """
    area = 3.14159 * radius**2
```

A. `int`

B. `float`

C. `str`

D. `char`

E. None of the above.

Question 17. [1 MARK]

Suppose we want to assign `True` to the name `validate` when a user has typed *only* the single letter `'a'`, `'b'`, `'c'`, or `'d'`. Which proposition should replace `#sub` to accomplish this?

```
value = input("Enter a single character: ")
validate = #sub
```

A. `value == "a" or "b" or "c" or "d"`

B. `value in "abcd"`

C. `value not in "efghijklmnopqrstuvwxyz"`

D. `value in ["a", "b", "c", "d"]`

E. More than one of the above.

Question 18. [1 MARK]

Which statement below is *false*?

A. Object oriented programming is no more powerful than imperative programming.

B. A constant value in Python *can* be modified.

C. We do not have to verify preconditions because it is the user's fault for breaking them.

D. Only *immutable* types can be keys in dictionaries.

E. All statements are true.

Question 19. [1 mark]

What tuple is assigned to `ans` when the following code is executed?

```
def foo(xs: tuple) -> tuple:
    a, b = xs
    a, b = b, a % b
    return (a, b)

ans = foo(foo((3,5)))
```

A. `(2, 2)`

B. `(2, 3)`

C. `(3, 2)`

D. `(3, 3)`

E. None of the above.

Question 20. [1 mark]

What can replace `#sub` so that `"Drake is overrated"` is assigned to `ys`?

```
xs = ["Drake", " ", "is", " ", "overrated"]
ys = #sub
```

A. `join(xs)`

B. `xs.join('')`

C. `''.join(xs)`

D. More than one of the above.

E. None of the above.

Question 21. [1 mark]

What is the most appropriate type hint (i.e. type contract) for the following?

```
def foo(x, y):
    for z in y:
        if (" "+z) in x:
            x[" "+z] += 1
    return sum(x[k] for k in x)
```

A. `def foo(x: dict[int, int], y: list[int]) -> int:`

B. `def foo(x: dict[int, str], y: list[int]) -> int:`

C. `def foo(x: dict[str, int], y: list[str]) -> int:`

D. `def foo(x: dict[str, int], y: str) -> int:`

E. None of the above.

Question 22. [1 MARK]

What type of error does the following code throw?

```
x = 0
if x > 0:
    y[x] = 1
else:
    y[0] = 1
```

A. `KeyError`

B. `IndexError`

C. `SyntaxError`

D. `NameError`

E. This is valid Python code.

Question 23. [1 MARK]

What is the value of a after the following code has been executed.

```
xss = [[1], [2,3], [1,2,3], [4,5,6]]
ys = [2, 3]
a = [ys[0] in xs and ys[1] in xs for xs in xss]
```

A. `[True]`

B. `[False]`

C. `[False, True, True, False]`

D. `[True, False, False, True]`

E. None of the above.

Question 24. [1 MARK]

What is stored in ans after the following is executed, supposing the user enters "25" and then "30".

```
x = input("Enter a number: ")
y = input("Enter a number: ")
ans = int(x + y)
```

A. `55`

B. `"55"`

C. `2530`

D. `"2530"`

E. `Error`

Question 25. [1 MARK]

What is assigned to ans after executing the following code?

```
def foo(x, y):
    if not x % 2:
        return foo(x//2, y-1)
    if not y:
        return x+y
    return foo(x-1, y-1)

ans = foo(3, 4)
```

A. 1

B. 2

C. 4

D. 7

E. `RecursionError`

Question 26. [1 MARK]

What is the value of `xs` after executing the following?

```
xs = [[1], [2, 3], [4, 5, 6]]
for x in xs:
    x.extend(x)
```

A. `[[1], [2, 3], [4, 5, 6]]`

B. `[[1], [1], [2, 3], [2, 3], [4, 5, 6], [4, 5, 6]]`

C. `[1, 1], [2, 3, 2, 3], [4, 5, 6, 4, 5, 6]]`

D. `Error`

E. None of the above.

Question 27. [1 MARK]

Which statement is *false*?

A. Every `if-then-else` statement can be written using *only* `if` statements.

B. Every `while` loop can be written as a `for` loop.

C. Functions can be defined inside of functions.

D. Every `for` loop can be written as a `while` loop.

E. More than one of the above.

Question 28. [1 MARK]

Which option *does not* assign the reverse of `xs` into `a`? Note: the reverse of `[1, 2, 3]` is `[3, 2, 1]`.

A. `a = xs[::-1]`

B. `a = xs.reverse()`

C. `a = xs[len(xs)::-1]`

D. `a = xs[-1::-1]`

E. None of the above.

Question 29. [1 MARK]

Consider the following function.

```
def foo(xss: list[list[int]]) -> bool:
    for k, xs in enumerate(xss):
        for x in xs:
            for ys in xss[:k]:
                if x in ys:
                    return False
            for ys in xss[k+1:]:
                if x in ys:
                    return False
    return True and len(xss) > 1
```

What best describes `foo`'s behaviour? Note: A list is said to be made up of *elements*.

A. `foo` returns `False` *only when* there are two distinct elements of `xss` that are equal.

B. `foo` returns `False` *only when* there are two distinct elements of `xss` that have a common element.

C. `foo` returns `True` *only when* there is an empty list in `xss`.

D. `foo` *always* returns `False`.

E. `foo` *always* returns `True`.

Question 30. [1 MARK]

What is the purpose of "getter" 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. None of the above.

Question 31. [1 MARK]

How many of the following expressions evaluate to `False`?

```
bool("")   # the empty string
bool(" ")  # space
bool(-1)
bool(0)
bool([])
```

A. 1

B. 2

C. 3

D. 4

E. 5

Question 32. [1 MARK]

Consider the following assignments.

```
xss = [[ 1,  2,  3,  4,  5,  6],
       [ 7,  8,  9, 10, 11, 12],
       [13, 14, 15, 16, 17, 18],
       [19, 20, 21, 22, 23, 24],
       [25, 26, 27, 28, 29, 30]]

yss = [[10, 11, 12],
       [16, 17, 18],
       [22, 23, 24]]
```

How many of the following expressions are equivalent to `yss`?

```
[xs[-3:] for xs in xss[1:4]]
[xs[3:] for xs in xss[1:-1]]
[xs[-3:] for xs in xss[1:4]]
[xs[3:] for xs in xss[-4:4]]
```

A. 0

B. 1

C. 2

D. 3

E. 4

Question 33. [1 MARK]

What is printed after executing the following?

```
x, stars = 1, '*'
while x <= 4:
    print(stars)
    stars *= 2
    x *= 2
```

A.
```
*
```

B.
```
*
**
```

C.
```
*
**
****
```

D.
```
*
**
****
********
```

E. Infinitely many stars.

Question 34. [1 MARK]

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

```
def validate() -> int:
    """
    Prompts the user to enter an integer.
    Repeats until a number is entered.
    """
    try:
        return int(input("Enter an integer "))
    except <Error>:
        return validate()
```

A. `ValueError`

B. `TypeError`

C. `NameError`

D. `InputError`

E. None of the above.

Question 35. [1 MARK]

Which type *cannot* be used as a *key* in a dictionary?

A. `int`

B. `str`

C. `list`

D. `tuple`

E. None of the above.

Question 36. [1 MARK]

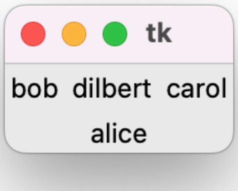
The following code generates the window to its right. What replaces #sub to generate it?

```
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()
```



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

B. (tk.BOTTOM, tk.RIGHT, tk.TOP, tk.BOTTOM)

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

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

E. None of the above.

The following *four questions* refer to the following class definition.

```
class A():
    def __init__(self, x: int) -> None:
        self._x = x
        self._y = 3

    def f(self, x: int) -> int:
        return 2*x

    def g(self, x: int) -> int:
        return self.f(self._y)

class B(A):
    def __init__(self, x: int, y: int) -> None:
        super().__init__(x)
        self._y = y

    def f(self, x: int) -> int:
        return A.f(self, self._y) + x

class C(B):
    def __init__(self, x: int) -> None:
        super().__init__(x, 5)

    def h(self) -> int:
        return self.g(self._y) + super().f(self._x)

class D(C):
    def __init__(self, x: int, y: int, z: int) -> None:
        super().__init__(x)
        self._y += 5
        self._z = z

    def g(self, x: int, y: int, z: int) -> int:
        return x*y*z + self._x + self._y + self._z

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

Question 37. [1 MARK]

What does `a.g(5)` return?

A. 2

B. 6

C. 10

D. `Error`

E. None of the above.

Question 38. [1 MARK]

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

A. 6

B. 8

C. 10

D. `Error`

E. None of the above.

Question 39. [1 MARK]

What does `c.h()` return?

A. 6

B. 15

C. 28

D. `Error`

E. None of the above.

Question 40. [1 MARK]

What does `d.g(1, 2, 3)` return?

A. 28

B. 32

C. 36

D. `Error`

E. None of the above.

END OF EXAMINATION