COMP 1405 Introduction to Computer Science I

Study Session Questions

December 26, 2025

1. If X = FALSE and Y = TRUE, which of these will return TRUE?

   ○ X and Y
   ○ not(X and Y)
   ○ (not X) AND (not Y)
   ○ X OR Y

2. Consider the following logic circuit. What is the output if X = True and Y = False?

   A. True
   B. False

3. Match each loop to its type.

   1.

```
run = True
while run:
    print("hello")
    run = False
```

   A. pre-condition with flag variable

   2.

```
x = 0
while True:
    x += 1
    if x > 2:
        break
```

   B. post-condition without flag variable

   3.

```
run = True
while True:
    if not run:
        break
    run = False
```

   C. post-condition with flag variable

   4.

```
x = 0
while x < 3:
    print("hello")
    x += 1
```

   D. pre-condition without flag variable

4. Match each control statement with what it does.

1. `break`	**A.** exits the current loop
2. `continue`	**B.** restarts the current loop
3. `pass`	**C.** doesn't do anything (placeholder)
4. `return`	**D.** exits the current function

5. Which of the following Python data types are mutable?

   ○ List
   ○ Tuple
   ○ Set
   ○ String
   ○ Dictionary
   ○ none of these

6. What Python keyword is used to define a function? ____________

7. Which of the following are invalid Python function names?

```
1 int_value = int(True)
2 int_value = int([1, 2, 3])
3 unk_value = "4.2.0"
4 int_value = int(" 42 ")
5 str_value = str(45.67)
6 string_value = 3 + "hello"
```

   A. 1 and 2
   B. 3 and 4
   C. 5 and 6
   D. 2 and 6

8. Which of the following are invalid Python function names?

```
3 def calculateSum(): return
4 def 2func(): return
5 def _private_func(): return
6 def sum-all(): return
```

   A. 3 and 4
   B. 4 and 6
   C. 4 only
   D. 6 only

9. Match each data type to their properties.

1. set	**A.** elements are u
2. dict	**B.** elements are st
3. list	**C.** elements are o
4. tuple	**D.** data is immuta

10. Which of the following statements are true? Select all that apply.

    ○ Declaring a variable in a def without global makes it local.

    ○ global variables cannot be accessed in nested defs.

    ○ Global vars can be accessed locally without a keyword if it isn't modified.

    ○ global variables are automatically immutable and cannot be changed.

    ○ A local variable declared in a def can't be accessed outside that def.

    ○ 'global' allows a def to modify a variable declared outside the def.

11. Match each string operation to the output `"mputhCompeci"`.

    1. `str.split(" ")[1] + str.split(" ")[0][1:4]`

    2. `str[:5] + str.split()[-1][:3]`

    3. `"-".join(str.split())[:12]`

    4. `str.split()[0][2:5] + str.split()[1][:2]`

12. Which of the following statements about lists in Python are TRUE? Select all that apply.

    ○ Lists are immutable.

    ○ Lists can contain elements of different types.

    ○ Lists must have unique elements.

    ○ Lists are unordered collections.

    ○ Lists do not support slicing.

    ○ All of these are false.

13. Match each list method to its description.

    1. `append()`

    2. `pop()`

    3. `remove()`

    4. `insert()`

    A. adds an eleme

    B. deletes the ele
    (or last by defaul

    C. deletes the firs

    D. adds an eleme

14. What will the following code output?

```
bob = [1, 2, 3]
def pat(krb):
    krb.append(4)

pat(bob)
print(bob)
```

    A. [1, 2, 3]

    B. [1, 2, 3, 4]

    C. Error: 'list' object has no attribute 'append'

    D. None

15. What is the value of `num` after `num = my_list[2][3]`?

```
my_list = [
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10, 11, 12]
]
```

A. 9

B. 10

C. 11

D. 12

16. Which of the following are true about exceptions in Python?

○ An except block can only handle one type of exception at a time.

○ The except block is executed when an exception is raised in the try block.

○ The try/except block can be nested.

○ The try block can have multiple except blocks.

○ The try block can handle exceptions even if there is no except block.

○ none of these are true

17. Match each error to its meaning.

1. Syntax Error

2. Runtime Error

3. Logic Error

A. a problem checked by t
incorrect use of the langua

B. a problem that isn't ch
interpreter but crashes the

C. a problem that doesn't
but produces unintended r

18. Which of the following are true of binary search? Select all that apply.

○ Binary search compares the target value with each element sequentially.

○ Binary search divides a list into halves and eliminates one half at a time.

○ Binary search can only be applied to sorted lists or arrays.

○ Binary search has a worst-case time complexity of $O(\log n)$.

○ Binary search has a worst-case time complexity of $O(n)$.

○ Binary search can be implemented recursively or iteratively.

19. Match each code line to what it does.

1. `open(file, 'r')`

2. `open(file, 'w')`

3. `open(file, 'a')`

4. `open(file, 'x')`

A. reads through the
the file does not exist

B. overwrites the cont
file if it does not exist

C. writes from the en
file if it does not exist

D. creates the specifie
the file exists

20. Which of the following are true of linear search? Select all that apply.

○ Linear search has a time complexity of $O(n)$ in the worst case.

○ Linear search compares each element of the list with the target in order.

○ Linear search requires the list to be in descending order.

○ Linear search is faster than binary search on sorted lists.

○ Linear search has a best-case time complexity of $O(\log n)$.

○ Linear search is efficient for small datasets.

21. What will the list `[8,5,4,3,7,6,1,0,9,2]` look like after 4 swaps using each of the following methods? Match accordingly.

1. Quick Sort (using the last element as a pivot)	**A.** `[0,1,2,3,4,6,5,8,9,7]`
2. Bubble Sort	**B.** `[5,4,3,7,8,6,1,0,9,2]`
3. Selection Sort (selecting the largest element)	**C.** `[2,5,4,3,0,1,6,7,8,9]`

22. Which adjacency list represents the following graph?

A. [[1, 5], [0, 2, 3, 5], [1, 4], [1, 4], [2, 3], [0, 1, 5]]

B. [[1, 5], [3, 5], [1, 4], [1, 4], [], [0, 5]]

C. [[1, 5], [3, 5], [1, 4], [1, 4], [3], [0]]

D. [[5], [0, 2, 3], [], [1], [2, 3], [0, 1, 5]]

23. Which adjacency matrix represents the following graph?

A. $\begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$

B. $\begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$

C. $\begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$

D. $\begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$

24. What will happen if a recursive function does not have a base case?

A. The computer will violently explode.

B. The program will throw a syntax error.

C. The program will crash with a stack overflow.

D. The program will return an incorrect result (logic error).

25. What will the following code output if the user enters `50`?

```
x = input("num:")
try:
    y = x * 2
    print(f"Result_1: {y}")
    z = int(x) + 2
    print(f"Result_2: {z}")
except ValueError:
    print("Error!")
```

A. Result_1: 100 Result_2: 52

B. Result_1: 5050 Result_2: 52

C. Error!

D. Result_1: 5050 Result_2: Error!

26. What will be printed when the following code runs?

```
a, b, c = 10, 25, 30
if a > b:
    if b > c:
        print("David")
    else:
        print("Mohammad")
else:
    if a + b > c:
        print("Eleena")
        if c - a == b:
            print("Jakob")
        else:
            print("Tiffany")
    else:
        print("Aayla")
```

A. David

B. Mohammad

C. Jakob

D. Aayla

27. What will the following code return?

```
aaa = 10
bbb = [5, 15]

def foo():
    aaa = 20
    bbb[0] = 50
    print(aaa)

def bar():
    global aaa
    aaa = 30
    bbb = [100, 200]
    return bbb

ccc = foo()
ddd = bar()
print(aaa, bbb, ccc, ddd)
```

A. 20 30 [50, 15] None [100, 200]

B. 20 10 [50, 15] 20 [100, 200]

C. 10 10 [50, 15] None [5, 15]

D. 20 30 [100, 200] 20 [100, 200]

28. Which of the following inputs (x, y) will make this code output "yippee"? Select all that apply.

```
x = int(input("num: "))
y = int(input("num: "))

if x > 0 and y > 0:
    print("yippee")
elif x < 0 and y < 0:
    print("no")
elif x > 0 or y > 0:
    if x == 0 or y == 0:
        print("no")
    else:
        print("yippee")
else:
    print("no")
```

○ (2, 4)

○ (2, -4)

○ (-2, 0)

○ (-2, -4)

○ (0, 0)

○ none of these

29. What will the following code output?

```
counter = 0
for i in range(6):
    if i % 2 == 0:
        continue
    for j in range(3):
        if j == 2:
            break
        counter += 1
        if i + j > 3:
            counter += 1
            break
print(counter)
```

A. 5

B. 6

C. 7

D. 8

E. 9

F. the code will not execute OR infinite loop

30. What will the following code output?

```
matrix = [
    [1, 2, 3, 4],
    [2, 1, 4, 3],
    [3, 4, 1, 2],
    [2, 3, 4, 1]
]
matrix[1].pop(2)
matrix[3].remove(matrix[2][2])
matrix.insert(2, [2, 4, 1, 3])
matrix[2].remove(3)
matrix.pop(1)
matrix[3].insert(1, matrix[2].pop(3))
print(matrix)
```

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

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

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

D. [[1,2,3,4], [2,4,1], [4,1,2], [2,3,4,1]]

31. What will the following code output?

```
names = {
    1: ['Isa', 'Sandra', 'Alik'],
    2: ['Kyle', 'Nolan', 'Tinaye'],
    3: ['Simon', 'Jon-Luca', 'Cindy']
}

result = []
for key, value in names.items():
    if len(value[0]) % key == 0:
        result.append(value[1])
    elif key + len(value[0]) > 5:
        result.append(value[2])
print(result)
```

A. ['Isa', 'Kyle', 'Nolan']

B. ['Jon-Luca', 'Sandra']

C. ['Sandra', 'Nolan', 'Cindy']

D. ['Alik', 'Tinaye', 'Jon-Luca']

32. What will the following code output?

```
def fac(n):
    if n == 0:
        return 1
    return n * fac(n - 1)

def fib(n):
    if n <= 1:
        return n
    return fac(n - 1) + fib(n - 2)

print(fib(5))
```

A. 7

B. 27

C. 48

D. 64

33. What will the following code output?

```
try:
    data = {"a": 1, "b": 2}
    value = 10 / 0
    value = data["c"]
except ZeroDivisionError as e:
    print(f"oops, {e}")
except LookupError as e:
    print(f"oopsie: {e}")
except Exception as e:
    print(f"oops again: {e}")
```

A. oops, division by zero

B. oops, division by zero oopsie: ’c’ oops again: ’c’

C. oops, division by zero oopsie: ’c’ oops again: division by zero

D. oops again: ’c’

34. What should we replace line 9 with to make the code function as intended?

```
class Robot:
    def __init__(self, model, task):
        self.model = model
        self.task = task

    def perform_task(self):
        return f"Robot {self.model} is performing: {self.task}"

# i’m on break, fix it yourself
print(worker.perform_task())
```

A. worker = new Robot("PhatGPT", "yapping")

B. worker = Robot("WALL-E", "garbage collection")

C. worker = new Robot("Sonny")

D. $worker = Robot.perform_task(”GLaDOS”, ”experiment”)$