Started on	Tuesday, 24 June 2025, 2:07 PM
State	Finished
Completed on	Tuesday, 24 June 2025, 2:34 PM
Time taken	27 mins 33 secs
Marks	17.00/25.00
Grade	**68.00** out of 100.00

Question 1

Complete

Mark 1.00 out of 1.00

(Predict the output)

```
List<Integer> list = List.of(1,2,3,4,5);
long even = list.parallelStream()
            .filter(i -> i % 2 == 0)
            .count();
System.out.println(even);
```

- a. Depends on thread scheduling
- b. 2
- c. 0
- d. 3

Question 2

Complete

Mark 1.00 out of 1.00

Which snippet prints

Found: 42

only if any element equals 42?

- a.
```
Stream.of(10,42,30)
    .filter(i->i==42)
    .findFirst()
    .ifPresent(i->System.out.println("Found: "+i));
```
- b.
```
Stream.of(10,42,30)
    .filter(i->i==42)
    .forEach(i->System.out.println("Found: "+i));
```
- c.
```
Stream.of(10,42,30)
    .findAny(i->i==42)
    .ifPresent(i->System.out.println("Found: "+i));
```
- d.
```
List.of(10,42,30).stream()
   .anyMatch(i->i==42)
   .ifPresent(i->System.out.println("Found: "+i));
```

Question 3

Complete

Mark 1.00 out of 1.00

Which snippet produces the product of all elements 24 from:

List<Integer> nums = List.of(1,2,3,4);

- a. nums.stream().reduce(1, (a,b)->a*b);
- b. nums.stream().reduce((a,b)->a*b).get();
- c. Both A and B
- d. Neither

Question 4

Complete

Mark 1.00 out of 1.00

Which snippet produces a sorted, distinct list of even squares [4,16,36] from:

List<Integer> nums = List.of(1,2,3,4,5,6);

- a.
```
nums.stream()
    .map(i->i*i)
    .filter(i->i%2==0)
    .sorted()
    .distinct()
    .collect(Collectors.toList());
```
- b.
```
nums.stream()
    .filter(i->i%2==0)
    .map(i->i*i)
    .distinct()
    .sorted()
    .toList();
```
- c. Both A and B
- d. Neither

Question 5

Complete

Mark 0.00 out of 1.00

(Predict the output)

```
Optional<String> first = Stream.<String>empty().findFirst();
System.out.println(first.isPresent());
```

- a. true
- b. null
- c. NoSuchElementException
- d. false

Question 6

Complete

Mark 0.00 out of 1.00

```
def make_funcs():
    return [lambda x: i * x for i in range(3)]

funcs = make_funcs()
results = [f(2) for f in funcs]
print(results)
```

What is the output?

- a. [4, 4, 4]
- b. [0, 1, 2]
- c. [0, 2, 4]
- d. [2, 4, 6]

Question 7

Complete

Mark 0.00 out of 1.00

```
def append_to_list(val, lst=[]):
    lst.append(val)
    return lst

print(append_to_list(1))
print(append_to_list(2))
```

What is the output?

- a. [1], [1, 2]
- b. [1, 2], [2]
- c. [2], [1, 2]
- d. [1], [2]

Question 8

Complete

Mark 1.00 out of 1.00

```
def star(func):
    def wrapper():
        return "*** " + func() + " ***"
    return wrapper

@star
def message():
    return "Hello"

print(message())
```

What is the output?

- a. *** message ***
- b. *** Hello ***
- c. SyntaxError
- d. Hello

Question 9

Complete

Mark 0.00 out of 1.00

```
def counter():
    yield 1
    yield 2
    yield 3

gen = counter()
print(next(gen))
print(list(gen))
```

What is the output?

- a. 1, [1, 2]
- b. 1, [2, 3]
- c. 1, [1]
- d. 1, [1, 2, 3]

Question 10

Complete

Mark 1.00 out of 1.00

```
nums = [1, 2, 3, 4]
  result = [x * x for x in nums if x % 2 == 0]
  print(result)
```

What is the output?

- a. [1, 9]
- b. [4, 16]
- c. [2, 4]
- d. [1, 4, 9, 16]

Question 11

Complete

Mark 1.00 out of 1.00

What is the output of the following SQL query?

```
SELECT name, salary,
    RANK() OVER (ORDER BY salary DESC) AS rank
FROM employees;
```

- a. Get employees sorted by name with ranking
- b. Assign dense rank based on salary
- c. Get top 5 salaries
- d. Assign rank to employees based on salary (highest = 1)

Question 12

Complete

Mark 1.00 out of 1.00

What does this SQL query return?

```
SELECT name
FROM employees e
WHERE salary > (
   SELECT AVG(salary)
   FROM employees
   WHERE department_id = e.department_id
);
```

- a. Selects all employees in the highest paid department
- b. Selects employees earning more than their department's average
- c. Selects employees with salaries above overall average
- d. Selects employees earning less than average salary

Question 13

Complete

Mark 1.00 out of 1.00

Which query returns employees with or without a department?

a. SELECT e.name, d.name
FROM departments d
LEFT JOIN employees e ON d.id = e.department_id;

b. SELECT e.name, d.name
FROM employees e
JOIN departments d ON e.department_id = d.id;

c. SELECT e.name, d.name
FROM employees e
LEFT JOIN departments d ON e.department_id = d.id;

d. SELECT e.name, d.name
FROM employees e
RIGHT JOIN departments d ON e.department_id = d.id;

Question 14

Complete

Mark 0.00 out of 1.00

Which of the following is TRUE about indexes in SQL databases?

a. Indexes speed up INSERT operations

b. Indexes are always automatically created

c. Composite indexes are useless in filtering

d. Indexes can slow down UPDATE and DELETE operations

Question 15

Complete

Mark 1.00 out of 1.00

What does this query return?

```
SELECT department_id, COUNT(*) as cnt
FROM employees
GROUP BY department_id
HAVING COUNT(*) > 5;
```

- a. Lists employees with salary > 5
- b. Lists departments with less than 5 employees
- c. Lists all departments
- d. Lists departments with more than 5 employees

Question 16

Complete

Mark 0.00 out of 1.00

Which of the following page replacement algorithms may lead to Belady's anomaly?

- a. Clock
- b. Optimal Page Replacement
- c. Least Recently Used (LRU)
- d. First-In First-Out (FIFO)

Question 17

Complete

Mark 1.00 out of 1.00

Which of the following conditions is not necessary for a deadlock to occur?

- a. Mutual Exclusion
- b. Hold and Wait
- c. Circular Wait
- d. Preemption

Question 18

Complete

Mark 0.00 out of 1.00

In which of the following scenarios is a context switch likely to happen?

- a. A process enters the critical section
- b. A process performs a non-blocking I/O operation
- c. A process completes execution of its time quantum in cooperative multitasking
- d. A higher-priority process becomes ready

Question 19

Complete

Mark 1.00 out of 1.00

Which of the following best describes the working set model in virtual memory management?

- a. All pages accessed during the entire lifetime of a process
- b. Pages swapped into secondary memory
- c. Set of pages actively used during the last N instructions
- d. Set of pages in the page table

Question 20

Complete

Mark 0.00 out of 1.00

In UNIX file systems, what happens when a hard link to a file is created?

- a. A new inode is created pointing to the same data
- b. The link count in the inode is incremented
- c. A symbolic pointer is created
- d. File content is copied into a new block

Question 21

Complete

Mark 1.00 out of 1.00

What is the output of the following code snippet?

```
def inorder(root):
    if root:
        inorder(root.left)
        print(root.val, end=' ')
        inorder(root.right)

class Node:
    def __init__(self, val):
        self.val = val
        self.left = None
        self.right = None

root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
inorder(root)
```

- a. 4 1 2 3
- b. 4 2 1 3
- c. 2 4 1 3
- d. 1 2 3 4

Question 22

Complete

Mark 1.00 out of 1.00

What does the following code return?

```
def is_balanced(s):
    stack = []
    for ch in s:
        if ch == '(':
            stack.append(ch)
        elif ch == ')':
            if not stack or stack.pop() != '(':
                return False
    return len(stack) == 0

print(is_balanced("(()())"))
```

- a. False
- b. Runtime Error
- c. True
- d. None

Question 23

Complete

Mark 1.00 out of 1.00

What is the time complexity of the following function?

```
def recursive_count( n ):
    if n <= 1:
        return 1
    return recursive_count(n-1) + recursive_count(n-2)
```

- a. O(log n)
- b. O(2^n)
- c. O(n)
- d. O(n^2)

Question 24

Complete

Mark 1.00 out of 1.00

What is the output of this graph-based code?

```
graph = {
    'A': ['B', 'C'],
    'B': ['D'],
    'C': ['E'],
    'D': [],
    'E': []
}

def dfs(node, visited):
    if node not in visited:
        visited.add(node)
        for neighbor in graph[node]:
            dfs(neighbor, visited)

visited = set()
dfs('A', visited)
print(visited)
```

a. {'A', 'B', 'C'}

b. {'A', 'C', 'E'}

c. {'A', 'B', 'C', 'D', 'E'}

d. {'A', 'B', 'D'}

Question 25

Complete

Mark 1.00 out of 1.00

What will this code output?

```
def first_non_repeating_char(s):
    count = {}
    for ch in s:
        count[ch] = count.get(ch, 0) + 1
    for ch in s:
        if count[ch] == 1:
            return ch
    return None

print(first_non_repeating_char("aabbcdc"))
```

a. a

b. d

c. b

d. c