Started on	Tuesday, 1 October 2024, 8:47 PM
State	Finished
Completed on	Tuesday, 1 October 2024, 9:00 PM
Time taken	12 mins 47 secs
Marks	5.00/10.00
Grade	**50.00** out of 100.00

Question **1**

Complete

Mark 1.00 out of 1.00

Consider the following sequence of SQL commands:

BEGIN TRANSACTION;

INSERT INTO Employees VALUES (101, 'John Doe', 'HR');

SAVEPOINT A;

UPDATE Employees SET department = 'Finance' WHERE employee_id = 101;

ROLLBACK TO A;

COMMIT;

What will be the final state of the Employees table after the transaction?

Select one or more:

- [ ] **a. An error will occur due to the use of SAVEPOINT**
- [ ] **b. No changes, as everything was rolled back**
- [x] **c. The employee 'John Doe' will be in the 'HR' department**
- [ ] **d. The employee 'John Doe' will be in the 'Finance' department**

Question 2

Complete

Mark 1.00 out of 1.00

Which of the following fills in the blank so that the code outputs one line but uses a poor

practice?

```
import java.util.*;
public class Cheater {
 int count = 0;
 public void sneak(Collection<String> coll) {
 coll.stream().
;
 }
 public static void main(String[] args) {
 Cheater c = new Cheater();
 c.sneak(Arrays.asList("weasel"));
 }
}
```

Select one or more:

- a. **peek(r -> System.out.println(r)).findFirst()**
- b. **peek(System.out::println).findFirst()**
- c. **peek(System.out::println)**
- d. **peek(r -> {count++; System.out.println(r); }).findFirst()**

Question 3

Complete

Mark 1.00 out of 1.00

Given the following sequence of operations on a circular queue:

enqueue(1), enqueue(2), enqueue(3), dequeue(), enqueue(4), enqueue(5), dequeue(), enqueue(6), what will the queue contain?

Select one or more:

- a. **2, 3, 5, 6**
- b. **3, 4, 5, 6**
- c. **4, 5, 6, 1**
- d. **1, 2, 4, 6**

Question 4

Complete

Mark 1.00 out of 1.00

Consider the following SQL commands:

BEGIN TRANSACTION;

UPDATE employee SET salary = 5000 WHERE emp_id = 101;

DELETE FROM employee WHERE emp_id = 102;

COMMIT;

DELETE FROM employee WHERE emp_id = 103;

ROLLBACK;

What will be the state of the employee table after executing the above commands?

Select one or more:

- [ ] **a. Both rows with emp_id = 102 and emp_id = 103 will be deleted**
- [ ] **b. No rows will be deleted from the table**
- [ ] **c. Only the row with emp_id = 103 will be deleted**
- [x] **d. The row with emp_id = 102 will be deleted, but the row with emp_id = 103 will remain**

Question 5

Complete

Mark 0.00 out of 1.00

In the context of the semaphore code given below, if sem_init(&mutex, 0, 0) was used instead of sem_init(&mutex, 0, 1), what would be the effect?

```
#include <semaphore.h>
#include <pthread.h>

sem_t mutex;

void* thread_function(void* arg) {
    sem_wait(&mutex);
    // Critical section
    printf("Thread %d in critical section\n", *((int*)arg));
    sem_post(&mutex);
    return NULL;
}

int main() {
    pthread_t t1, t2;
    int t1_id = 1, t2_id = 2;
    sem_init(&mutex, 0, 1);

    pthread_create(&t1, NULL, thread_function, (void*)&t1_id);
    pthread_create(&t2, NULL, thread_function, (void*)&t2_id);

    pthread_join(t1, NULL);
    pthread_join(t2, NULL);

    sem_destroy(&mutex);
    return 0;
}
```

Select one or more:

- [ ] **a. The semaphore initialization would fail, and the program would not compile**
- [ ] **b. Both threads would enter the critical section simultaneously**
- [x] **c. One thread would enter the critical section, but the other would never be able to enter**
- [ ] **d. The critical section would never be accessed by any thread**

Question 6

Complete

Mark 1.00 out of 1.00

In Java, the Queue interface provides a method poll(). What is the key difference between poll() and remove() when operating on a queue?

Select one or more:

- [x] **a. poll() returns null if the queue is empty, while remove() throws an exception**
- [ ] **b. poll() adds an element to the front of the queue, while remove() adds it to the rear**
- [ ] **c. poll() and remove() perform exactly the same function**
- [ ] **d. poll() throws an exception if the queue is empty, while remove() returns null**

Question 7

Complete

Mark 0.00 out of 1.00

What is the output of the following application?

```
package holiday;
enum DaysOff {
 Thanksgiving, PresidentsDay, ValentinesDay
}
public class Vacation {
 public static void main(String... unused) {
 final DaysOff input = DaysOff.Thanksgiving;
 switch(input) {
 default:
 case DaysOff.ValentinesDay:
 System.out.print("1");
 case DaysOff.PresidentsDay:
 System.out.print("2");
 }
 }
}
```

Select one or more:

- [ ] **a. 1**
- [ ] **b. None of the above**
- [x] **c. 12**
- [ ] **d. 2**

Question 8

Complete

Mark 0.00 out of 1.00

What is the output of the following application?

```
package beach;
import java.util.function.*;
class Tourist {
 public Tourist(double distance) {
 this.distance = distance;
 }
 public double distance;
}
public class Lifeguard {
 private void saveLife(Predicate<Tourist> canSave, Tourist tourist) {
 System.out.print(canSave.test(tourist) ? "Saved" : "Too far"); // y1
 }
 public final static void main(String... sand) {
 new Lifeguard().saveLife(s -> s.distance<4, new Tourist(2)); // y2
 }
}
```

Select one or more:

- [x] **a. The code does not compile because of line y1.**
- [ ] **b. The code does not compile because of line y2.**
- [ ] **c. Saved**
- [ ] **d. Too far**

Question 9

Complete

Mark 0.00 out of 1.00

Consider the following pseudocode for two processes using a shared buffer and semaphores:

```
semaphore empty = 10;  // Number of empty slots in the buffer
semaphore full = 0;    // Number of filled slots in the buffer
semaphore mutex = 1;   // For mutual exclusion

Process A: // Producer
   while (true) {
      produce_item();
      wait(empty);
      wait(mutex);
      add_item_to_buffer();
      signal(mutex);
      signal(full);
   }

Process B: // Consumer
   while (true) {
      wait(full);
      wait(mutex);
      remove_item_from_buffer();
      signal(mutex);
      signal(empty);
      consume_item();
   }
```

In this producer-consumer problem, what will happen if wait(mutex) is omitted from the producer and consumer code?

Select one or more:

- [x] **a. Multiple processes will try to access the buffer simultaneously, leading to race conditions**
- [ ] **b. Both producer and consumer processes will terminate**
- [ ] **c. The program will run without errors, as wait(mutex) is unnecessary**
- [x] **d. The buffer will become full and cause a deadlock**

Question **10**

Complete

Mark 0.00 out of 1.00

What is the output of the following application?

```
package park;
class LostBallException extends Exception {}
public class Ball {
 public void toss() throw LostBallException {
 throw new ArrayStoreException();
 }
 public static void main(String[] bouncy) {
 try {
 new Ball().toss();
 } catch (Throwable e) {
 System.out.print("Caught!");
 }
 }
}
```

Select one or more:

- [ ] **a. The code does not compile because LostBallException is not handled or declared in the main() method.**
- [x] **b. The code does not compile because ArrayStoreException is not handled or declared in the toss() method.**
- [ ] **c. Caught!**
- [ ] **d. The code does not compile for a different reason.**