IT Skill Test GIC Myanmar

Duration: 30 Minutes   Total Questions: 8

---

1. In a system that manages employee records, you need to implement a method to validate email addresses. Which of the following implementations is the most appropriate for a basic email validation?

```
A.public boolean isValidEmail(String email) {
      return email.contains("@");
  }
B.public boolean isValidEmail(String email) {
      return email.matches("^[¥w-¥.]+@([¥w-]+¥.)+[¥w-]{2,4}$");
  }
C.public boolean isValidEmail(String email) {
      return email.length() > 5;
  }
D.public boolean isValidEmail(String email) {
      return email.endsWith(".com");
  }
```

2. You're developing a method to process an array of integers. The method should return the sum of all even numbers in the array. Which of the following implementations is correct?

```
A.public int sumEvenNumbers(int[] numbers) {
      int sum = 0;
      for (int num : numbers) {
          if (num % 2 == 0) {
              sum += num;
          }
      }
      return sum;
```

```
   }
```

```
B.public int sumEvenNumbers(int[] numbers) {
      int sum = 0;
      for (int i = 0; i < numbers.length; i += 2) {
         sum += numbers[i];
      }
      return sum;
   }
```

```
C.public int sumEvenNumbers(int[] numbers) {
      int sum = 0;
      for (int num : numbers) {
         if (num % 2 != 0) {
            sum += num;
         }
      }
      return sum;
   }
```

```
D.public int sumEvenNumbers(int[] numbers) {
      return Arrays.stream(numbers).sum();
   }
```

3. You're developing a method to check if a given year is a leap year. Which of the following implementations is correct?

```
A.public boolean isLeapYear(int year) {
      return year % 4 == 0;
   }
```

```
B.public boolean isLeapYear(int year) {
      return year % 400 == 0 || (year % 4 == 0 && year % 100 != 0);
   }
```

```
C.public boolean isLeapYear(int year) {
      return year % 100 == 0;
   }
```

```
D.public boolean isLeapYear(int year) {
        return year % 4 == 0 && year % 100 == 0;
    }
```

4. In a multi-threaded application, you need to implement a counter that can be safely incremented by multiple threads. Which of the following approaches is thread-safe?

```
A.private int counter = 0;
  public void increment() {
      counter++;
  }
B.private AtomicInteger counter = new AtomicInteger(0);
  public void increment() {
      counter.incrementAndGet();
  }
C.private int counter = 0;
  public synchronized void increment() {
      counter++;
  }
D.private volatile int counter = 0;
  public void increment() {
      counter++;
  }
```

5. You're developing a method to find the maximum value in an array of integers. Which of the following implementations is correct and efficient?

```
A.public int findMax(int[] arr) {
      return Arrays.stream(arr).max().getAsInt();
  }
B.public int findMax(int[] arr) {
      int max = arr[0];
      for (int i = 1; i < arr.length; i++) {
          if (arr[i] > max) {
              max = arr[i];
```

```
            }
        }
        return max;
    }
```

C.public int findMax(int[] arr) {

```
        Arrays.sort(arr);
        return arr[arr.length - 1];
    }
```

D.public int findMax(int[] arr) {

```
        return Collections.max(Arrays.asList(arr));
    }
```

6. You're working on a legacy system that uses the following method to validate user passwords. The security team has identified this as a potential vulnerability. What is the primary issue with this implementation?

```
public boolean isPasswordValid(String password) {
    if (password.length() < 8) {
        return false;
    }
    boolean hasUpperCase = false;
    boolean hasLowerCase = false;
    boolean hasDigit = false;
    for (char c : password.toCharArray()) {
        if (Character.isUpperCase(c)) {
            hasUpperCase = true;
        } else if (Character.isLowerCase(c)) {
            hasLowerCase = true;
        } else if (Character.isDigit(c)) {
            hasDigit = true;
        }
        if (hasUpperCase && hasLowerCase && hasDigit) {
            return true;
        }
    }
    return false;
}
```

A.The method doesn't check for special characters

B.The loop continues unnecessarily after all conditions are met

C.The method returns true as soon as one uppercase, one lowercase, and one digit are found

D.The method doesn't handle null password input

7. You're developing a method to simulate a dice game where a player rolls until they get a 6 or have rolled 3 times. Which looping construct would be most appropriate?

A.for (int i = 0; i < 3; i++) { ... }

B.while (rollCount < 3 && diceValue != 6) { ... }

C.do { ... } while (rollCount < 3 && diceValue != 6);

D.IntStream.range(0, 3).forEach(i -> { ... });

8. You're implementing a retry mechanism for a network operation that may fail. The operation should be attempted up to 5 times with increasing delays between attempts. Which looping construct would be most suitable for this implementation?

A.while (retryCount < 5) { ... }

B.do { ... } while (retryCount < 5);

C.for (int i = 0; i < 5; i++) { ... }

D.for (Attempt attempt : attempts) { ... }