A)

```
def compute_average(numbers):
    total = 0
    for number in numbers:
        total += number
    return total / len(numbers)
print("Average of [10, 20, 30]:", compute_average([10, 20, 30]))
print("Average of []:", compute_average([]))
```

B)

```
def compute_average(numbers):
    total = 0
    for i in range(len(numbers) + 1):
        total += numbers[i]  # When i == len(numbers), this causes an IndexError.
    return total / len(numbers)
print("Average of [5, 15, 25]:", compute_average([5, 15, 25]))
```

C)

```
def compute_average(numbers):
    total = 1
    for number in numbers:
        total *= number
    return total / len(numbers)
print("Average of [2, 4, 6]:", compute_average([2, 4, 6]))
```

D)

```
def compute_average(numbers):
    if not numbers:
        raise ValueError("List is empty, cannot compute average.")
    total = sum(numbers)
    return total / len(numbers)
```

```
def test_compute_average():
    assert compute_average([10, 20, 30]) == 20, "Average should be 20"
    try:
        compute_average([])
    except ValueError:
        pass
    else:
        raise AssertionError("Expected ValueError for empty list")
test_compute_average()
```

E)

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        reverse(arr);
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }

    public static void reverse(int[] arr) {
        for (int i = 0; i < arr.length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - i - 1];
            arr[arr.length - i - 1] = temp;
        }
    }
}
```

F)

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        reverse(arr);
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }

    public static void reverse(int[] arr) {
        for (int i = 0; i < arr.length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - i];.
            arr[arr.length - i] = temp;
        }
    }
}
```

G)

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        reverse(arr);
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
    public static void reverse(int[] arr) {
        for (int i = 0; i < arr.length / 2; i++) {
            arr[i] = arr[arr.length - i - 1];
            arr[arr.length - i - 1] = arr[i];
        }
    }
}
```

H)

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        reverse(arr);
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }

    public static void reverse(int[] arr) {
        for (int i = 0; i <= arr.length; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - i - 1];
            arr[arr.length - i - 1] = temp;
        }
    }
}
```

I)

```
def find_max(numbers):
    # Problem: Fails when the list is empty.
    max_val = numbers[0]
    for num in numbers:
        if num > max_val:
            max_val = num
    return max_val

print(find_max([]))  # Raises IndexError
```

J)

```
def find_max(numbers):
    if not numbers:
        raise ValueError("Some error message")
    max_val = numbers[0]
    for num in numbers[1:]:
        if num > max_val:
            max_val = num
    return max_val
```

K)

```
def find_max(numbers):
    max_val = 0
    for num in numbers:
        if num > max_val:
            max_val = num
    return max_val
print(find_max([-5, -2, -8]))
```

L)

```
def find_max(numbers):
    return max(numbers, default=None)
print(find_max([]))
```

M)

```
function isPalindrome(str) {
    for (let i = 0; i < str.length; i++) {
        if (str[i] !== str[str.length - 1 - i]) {
            return false;
        }
    }
    return true;
}
console.log(isPalindrome("Racecar"));
```

N)

```
function isPalindrome(str) {
    for (let i = 0; i <= str.length; i++) {
        if (str[i] !== str[str.length - 1 - i]) {
            return false;
        }
    }
    return true;
}
console.log(isPalindrome("level"));
```

O)

```
function isPalindrome(str) {
    let mid = Math.floor(str.length / 2);
    for (let i = 0; i < mid - 1; i++) {
        if (str[i] !== str[str.length - 1 - i]) {
            return false;
        }
    }
    return true;
}
console.log(isPalindrome("madam"));
```

P)

```
function isPalindrome(str) {
    const normalized = str.toLowerCase();
    const len = normalized.length;
    for (let i = 0; i < Math.floor(len / 2); i++) {
        if (normalized[i] !== normalized[len - 1 - i]) {
            return false;
        }
    }
    return true;
}
```