Assessment 1

CODE:

```
#!/bin/bash

# Ask for number of elements in array 1
echo "How many elements in array 1?"
read ele1

if [ "$ele1" -le 0 ]; then
    echo "Please enter at least one element."
    exit 1
fi

# Read elements into array 1
arr1=()
i=0
while [ $i -lt $ele1 ]; do
    echo "Enter element $((i+1)) for array 1:"
    read val
    arr1+=("$val")
    i=$((i+1))
done

# Ask for number of elements in array 2
echo "How many elements in array 2?"
read ele2

if [ "$ele2" -le 0 ]; then
    echo "Please enter at least one element."
    exit 1
fi

# Read elements into array 2
arr2=()
i=0
while [ $i -lt $ele2 ]; do
    echo "Enter element $((i+1)) for array 2:"
    read val
    arr2+=("$val")
```

```
    i=$((i+1))
done

# Function to check if a number is a palindrome
is_palindrome() {
    num="$1"
    rev=$(echo "$num" | rev)
    if [ "$num" = "$rev" ]; then
        return 0
    else
        return 1
    fi
}

# Function to check if a number is prime
is_prime() {
    num="$1"
    if [ "$num" -lt 2 ]; then
        return 1
    fi
    i=2
    while [ $i -lt "$num" ]; do
        if [ $((num % i)) -eq 0 ]; then
            return 1
        fi
        i=$((i+1))
    done
    return 0
}

# Function to calculate sum of digits
sum_of_digits() {
    num="$1"
    sum=0
    for (( i=0; i<${#num}; i++ )); do
        digit="${num:$i:1}"
        sum=$((sum + digit))
    done
    echo "$sum"
}

# Find palindromes from array 1
pal_arr=()
```

```
for val in "${arr1[@]}"; do
    is_palindrome "$val"
    if [ $? -eq 0 ]; then
        pal_arr+=("$val")
    fi
done

echo "Palindrome numbers: ${pal_arr[@]}"

# Find primes from array 2
prime_arr=()
for val in "${arr2[@]}"; do
    is_prime "$val"
    if [ $? -eq 0 ]; then
        prime_arr+=("$val")
    fi
done

echo "Prime numbers: ${prime_arr[@]}"

# Multiply each prime with sum of digits of each palindrome
final_arr=()
for p in "${pal_arr[@]}"; do
    digit_sum=$(sum_of_digits "$p")
    for q in "${prime_arr[@]}"; do
        result=$((digit_sum * q))
        final_arr+=("$result")
    done
done

echo "Final array (prime × sum of palindrome digits): ${final_arr[@]}"
```

👷 Line-by-Line Explanation

```
#!/bin/bash
```

- This tells the system to run the script using Bash.

```
echo "How many elements in array 1?"
read ele1
```

- Asks the user how many elements they want in the first array.
- Stores that number in the variable `ele1`.

```
if [ "$ele1" -le 0 ]; then
    echo "Please enter at least one element."
    exit 1
fi
```

- Checks if the number is less than or equal to 0.
- If yes, shows an error and stops the script.

```
arr1=()
i=0
while [ $i -lt $ele1 ]; do
    echo "Enter element $((i+1)) for array 1:"
    read val
    arr1+=("$val")
    i=$((i+1))
done
```

- Creates an empty array `arr1`.
- Uses a `while` loop to read `ele1` number of values from the user.
- Adds each value to `arr1`.

```
echo "How many elements in array 2?"
read ele2
```

- Asks how many elements to put in the second array.
- Stores the number in `ele2`.

```
if [ "$ele2" -le 0 ]; then
    echo "Please enter at least one element."
    exit 1
fi
```

- Checks if the number is valid.
- If not, shows an error and exits.

```
arr2=()
i=0
while [ $i -lt $ele2 ]; do
    echo "Enter element $((i+1)) for array 2:"
    read val
    arr2+=("$val")
    i=$((i+1))
done
```

- Creates an empty array `arr2`.
- Reads values from the user and stores them in `arr2`.

```
is_palindrome() {
    num="$1"
    rev=$(echo "$num" | rev)
    if [ "$num" = "$rev" ]; then
        return 0
    else
        return 1
    fi
}
```

- Defines a function to check if a number is a **palindrome**.
- Reverses the number and compares it to the original.
- Returns 0 if it's a palindrome, 1 if not.

```
is_prime() {
    num="$1"
    if [ "$num" -lt 2 ]; then
        return 1
    fi
    i=2
    while [ $i -lt "$num" ]; do
        if [ $((num % i)) -eq 0 ]; then
            return 1
        fi
        i=$((i+1))
    done
    return 0
}
```

- Defines a function to check if a number is **prime**.
- Tries dividing the number by all smaller numbers.
- If divisible, it's not prime.

```
pal_arr=()
for val in "${arr1[@]}"; do
    is_palindrome "$val"
    if [ $? -eq 0 ]; then
        pal_arr+=("$val")
```

```
    fi
done
```

- Creates an empty array `pal_arr`.
- Loops through `arr1` and adds only palindromes to `pal_arr`.

```
echo "Palindrome numbers: ${pal_arr[@]}"
```

- Prints all palindrome numbers found.

```
prime_arr=()
for val in "${arr2[@]}"; do
    is_prime "$val"
    if [ $? -eq 0 ]; then
        prime_arr+=("$val")
    fi
done
```

- Creates an empty array `prime_arr`.
- Loops through `arr2` and adds only prime numbers to `prime_arr`.

```
echo "Prime numbers: ${prime_arr[@]}"
```

- Prints all prime numbers found.

```
final_arr=()
for p in "${pal_arr[@]}"; do
    for q in "${prime_arr[@]}"; do
        result=$((p * q))
        final_arr+=("$result")
    done
done
```

- Creates an empty array `final_arr`.
- Multiplies each palindrome with each prime.
- Stores the result in `final_arr`.

```
echo "Final array (palindrome × prime): ${final_arr[@]}"
```

- Prints the final array of multiplied values.

Output of the above code:

```
vboxuser@ubuntu:~/Documents/Linux_Lab$ ./Assm.sh
How many elements in array 1?
3
Enter element 1 for array 1:
123
Enter element 2 for array 1:
33
Enter element 3 for array 1:
44
How many elements in array 2?
3
Enter element 1 for array 2:
12
Enter element 2 for array 2:
32
Enter element 3 for array 2:
23
Palindrome numbers: 33 44
Prime numbers: 23
Final array (prime × sum of palindrome digits): 138 184
vboxuser@ubuntu:~/Documents/Linux_Lab$
```