1.Check if the given number is Happy Number

```
def is_happy_number(n):
    # A set to store the numbers we've seen to detect cycles
    seen_numbers = set()

    while n != 1:
        # Calculate the sum of the squares of the digits
        n = sum(int(digit) ** 2 for digit in str(n))

        # If we have already seen this number, it means we've entered a cycle
        if n in seen_numbers:
            return False

        # Add the number to the set of seen numbers
        seen_numbers.add(n)

    return True

# Default input number for testing
number = 19  # Default test number for Happy Number
if is_happy_number(number):
    print(f"{number} is a Happy Number.")
else:
    print(f"{number} is not a Happy Number.")
```

Expected Output:

19 is a Happy Number.

2. check whether given string is a palindrome or not

```
def is_palindrome(s):
    # Remove spaces and convert the string to lowercase for case-insensitive comparison
```

```
    s = s.replace(" ", "").lower()

    # Compare the string with its reverse
    return s == s[::-1]

# Default input string for testing
input_string = "A man a plan a canal Panama"  # Default test string

# Check if the string is a palindrome
if is_palindrome(input_string):
    print(f"'{input_string}' is a palindrome.")
else:
    print(f"'{input_string}' is not a palindrome.")
```

Expected Output:

'A man a plan a canal Panama' is a palindrome.

3. sort the elements of an array in ascending order

```
import java.util.Arrays;

public class SortArray {
    public static void main(String[] args) {
        int[] arr = {12, 5, 7, 9, 1, 15};

        // Sorting the array in ascending order using Arrays.sort()
        Arrays.sort(arr);

        // Displaying the sorted array
        System.out.println("Sorted Array: " + Arrays.toString(arr));
    }
}
```

Expected Output:

Sorted Array: [1, 5, 7, 9, 12, 15]

4. Remove duplicates from a list

```
import java.util.*;

public class RemoveDuplicates {
    public static void main(String[] args) {
        // Create a list with some duplicate elements
        List<Integer> list = new ArrayList<>(Arrays.asList(1, 2, 3, 4, 5, 3, 2, 6, 7, 5));

        // Print the original list
        System.out.println("Original List: " + list);

        // Remove duplicates using a HashSet
        Set<Integer> set = new HashSet<>(list);

        // Convert the set back to a list to retain the original order
        list = new ArrayList<>(set);

        // Print the list after removing duplicates
        System.out.println("List after removing duplicates: " + list);
    }
}
```

Expected Output:

Original List: [1, 2, 3, 4, 5, 3, 2, 6, 7, 5]

List after removing duplicates: [1, 2, 3, 4, 5, 6, 7]

5. print the elements of an array present on even position

```
#include <stdio.h>
```

```
int main() {
    int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9};
    int n = sizeof(arr) / sizeof(arr[0]);

    printf("Elements at even positions:\n");

    // Loop through the array and print elements at even indices
    for (int i = 0; i < n; i++) {
        if (i % 2 == 0) {  // Checking for even index (starting from 0)
            printf("%d ", arr[i]);
        }
    }

    return 0;
}
```

Expected Output:

```
Elements at even positions:
1 3 5 7 9
```

6. Print the First N Natural Numbers and Calculate Their Sum Using Recursion

```
#include <stdio.h>

void print_natural_numbers(int n, int current, int total) {
    if (current > n) {
        printf("Sum of the first %d natural numbers is: %d\n", n, total);
        return;
    }
    printf("%d ", current);
    print_natural_numbers(n, current + 1, total + current);
```

```
}

int main() {
    int n = 10;  // Change this value to print the first N natural numbers
    print_natural_numbers(n, 1, 0);
    return 0;
}
```

Expected Output:

1 2 3 4 5 6 7 8 9 10 Sum of the first 10 natural numbers is: 55