

```
1)
def is_power_of_two(n):
    if n <= 0:
        return False

    while n % 2 == 0:
        n //= 2

    return n == 1

print(is_power_of_two(1)) # Output: True
print(is_power_of_two(16)) # Output: True
print(is_power_of_two(3)) # Output: False
```

```
2)
def sum_of_natural_numbers(n):
    return (n * (n + 1)) // 2

print(sum_of_natural_numbers(3)) # Output: 6
print(sum_of_natural_numbers(5)) # Output: 15
```

```
3)
def factorial(n):
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

print(factorial(5)) # Output: 120
print(factorial(4)) # Output: 24
```

```
4)
def exponentiate(N, P):
    return N ** P

print(exponentiate(5, 2)) # Output: 25
print(exponentiate(2, 5)) # Output: 32
```

```
5)
def find_maximum(arr):
    if len(arr) == 1:
        return arr[0]
    else:
        return max(arr[0], find_maximum(arr[1:]))

arr1 = [1, 4, 3, -5, -4, 8, 6]
print(find_maximum(arr1)) # Output: 8

arr2 = [1, 4, 45, 6, 10, -8]
```

```
print(find_maximum(arr2)) # Output: 45
```

6)

```
def find_nth_term(a, d, N):  
    return a + (N - 1) * d
```

```
print(find_nth_term(2, 1, 5)) # Output: 6
```

```
print(find_nth_term(5, 2, 10)) # Output: 23
```

7)

```
def permute(s, l, r):  
    if l == r:  
        print("".join(s))  
    else:  
        for i in range(l, r + 1):  
            s[l], s[i] = s[i], s[l] # Swap characters  
            permute(s, l + 1, r)    # Recursion  
            s[l], s[i] = s[i], s[l] # Backtrack, undo the swap
```

```
# Function to print all permutations of a given string
```

```
def print_permutations(string):  
    n = len(string)  
    s = list(string) # Convert string to a list of characters  
    permute(s, 0, n - 1)
```

```
# Test Example 1
```

```
print_permutations("ABC")
```

```
# Test Example 2
```

```
print_permutations("XY")
```

8)

```
def permute(s, l, r):  
    if l == r:  
        print("".join(s))  
    else:  
        for i in range(l, r + 1):  
            s[l], s[i] = s[i], s[l] # Swap characters  
            permute(s, l + 1, r)    # Recursion  
            s[l], s[i] = s[i], s[l] # Undo the swap
```

```
# Function to print all permutations of a given string
```

```
def print_permutations(string):  
    n = len(string)  
    s = list(string) # Convert string to a list of characters  
    permute(s, 0, n - 1)
```

```
# Test Example 1
```

```
print_permutations("ABC")
```

```
# Test Example 2
```

```
print_permutations("XY")
```