1)

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

        logarithm = math.log10(n) / math.log10(3)
        return math.floor(logarithm) == logarithm

import math

print(isPowerOfThree(27))  # Output: True
print(isPowerOfThree(0))   # Output: False
print(isPowerOfThree(-1))  # Output: False
```

2)

```
def lastRemaining(arr):
        return lastRemainingHelper(arr, 0, len(arr) - 1)

def lastRemainingHelper(arr, left, right):
        if left == right:
        return arr[left]

        size = right - left + 1

        if size % 2 == 0:
        return lastRemainingHelper(arr, left + 2, right - 1)
        else:
        return lastRemainingHelper(arr, left + 2, right - 2)

print(lastRemaining([1, 2, 3, 4, 5, 6, 7, 8, 9]))  # Output: 6
print(lastRemaining([1]))  # Output: 1
```

3)

```
def printSubsets(set_str, subset="", index=0):
        if index == len(set_str):
        print(subset)
        return

        printSubsets(set_str, subset + set_str[index], index + 1)
        printSubsets(set_str, subset, index + 1)
```

4)

```
def calculateLength(str, index=0):
        if index == len(str):
        return 0

        return 1 + calculateLength(str, index + 1)
```

5)

```
def countSubstrings(S):
        count = 0
        n = len(S)

        for i in range(n):
        for j in range(i, n):
        if S[i] == S[j]:
                count += 1

        return count

print(countSubstrings("abcab"))  # Output: 7
print(countSubstrings("aba"))  # Output: 4
```

6)

```
def towerOfHanoi(n, source, destination, auxiliary):
        if n == 1:
        print(f"move disk 1 from rod {source} to rod {destination}")
        return 1

        moves = 0

        moves += towerOfHanoi(n - 1, source, auxiliary, destination)
        print(f"move disk {n} from rod {source} to rod {destination}")
        moves += 1
        moves += towerOfHanoi(n - 1, auxiliary, destination, source)

        return moves
```

7)

```
def permutations(str):
        permutationsHelper(str, 0, len(str) - 1)


def permutationsHelper(str, left, right):
        if left == right:
        print(str)
        else:
        for i in range(left, right + 1):
        str = swap(str, left, i)
        permutationsHelper(str, left + 1, right)
        str = swap(str, left, i)  # Backtrack


def swap(str, i, j):
        charList = list(str)
        charList[i], charList[j] = charList[j], charList[i]
```

```
        return ''.join(charList)

8)
def countConsonants(string):
        count = 0
        vowels = "aeiou"

        for ch in string.lower():
        if ch.isalpha() and ch not in vowels:
        count += 1

        return count
```