1. Fibonacci Series

Python
```
def fibonacci(n):
  """
  This function recursively calculates the nth Fibonacci number.
  """
  if n < 0:
    print("Incorrect input")
    return
  elif n == 0:
    return 0
  elif n == 1:
    return 1
  else:
    return fibonacci(n-1) + fibonacci(n-2)

# Example usage:
for i in range(10):
  print(fibonacci(i), end=" ")
```

2. Armstrong Number

Python
```
def is_armstrong(number):
  original_number = number
  sum = 0
  num_of_digits = len(str(number))
  while number > 0:
    digit = number % 10
    sum += digit ** num_of_digits
    number //= 10
  return sum == original_number

# Example usage:
num = 153
if is_armstrong(num):
  print(f"{num} is an Armstrong number")
else:
  print(f"{num} is not an Armstrong number")
```

3. Greatest Common Divisor (GCD)

Python
```
def gcd(a, b):
  """
  This function recursively calculates the greatest common divisor (GCD) of
two numbers.
  """
  if b == 0:
    return a
  else:
    return gcd(b, a % b)

# Example usage:
x = 30
y = 45
gcd_value = gcd(x, y)
print(f"GCD of {x} and {y} is {gcd_value}")
```

4. Largest Element in Array

Python
```
def largest_element(arr, i):
  if i == len(arr) - 1:
    return arr[i]
  else:
    largest = largest_element(arr, i + 1)
    return largest if largest > arr[i] else arr[i]

arr = [10, 25, 12, 3, 70]
largest = largest_element(arr, 0)
print(f"Largest element in the array is {largest}")
```

5. Factorial

Python
```
def factorial(n):
  """
  This function recursively calculates the factorial of a number.
  """
  if n == 0:
    return 1
  else:
    return n * factorial(n-1)

# Example usage:
num = 5
fact = factorial(num)
print(f"Factorial of {num} is {fact}")
```

6. String Copy

Python
```
def copy_string(source, dest, i):
  """
  This function recursively copies a string to another string.
  """
  if source[i] == '\0':
    return
  else:
    dest[i] = source[i]
    copy_string(source, dest, i + 1)

# Example usage:
source_str = "Hello"
dest_str = [None] * len(source_str) + ["\0"]  # Create destination with
null terminator
copy_string(source_str, dest_str, 0)
print(f"Copied string: {''.join(dest_str[:-1])}")  # Remove null terminator
from output
```

7. String Reverse

Python
```
def reverse_string(string, i):
  """
  This function recursively reverses a string.
  """
  if i == len(string) // 2:
    return
  else:
    temp = string[i]
    string[i] = string[len(string) - i - 1]
    string[len(string) - i - 1] = temp
    reverse_string(string, i + 1)

# Example usage:
text = "World"
reverse_string(text, 0)
print(f"Reversed string: {text}")
```

8.Prime Numbers

```
def sieve_of_eratosthenes(n):
  """
  This function uses the Sieve of Eratosthenes to generate prime numbers up
to n.
  """
  primes = [True] * (n + 1)
  primes[0] = primes[1] = False  # 0 and 1 are not prime
  for i in range(2, int(n**0.5) + 1):
    if primes[i]:
      for j in range(i * i, n + 1, i):
        primes[j] = False
  return [i for i, is_prime in enumerate(primes) if is_prime]

# Example usage:
limit = 20
primes = sieve_of_eratosthenes(limit)
print(f"Prime numbers up to {limit}: {primes}")
```

9. Check Prime Number

Python
```
def is_prime(n):
  """
  This function recursively checks if a number is prime.
  """
  if n <= 1:
    return False
  elif n <= 3:
    return True
  elif n % 2 == 0 or n % 3 == 0:
    return False
  i = 5
  while i * i <= n:
    if n % i == 0 or n % (i + 2) == 0:
      return False
    i += 6
  return True

# Example usage:
num = 11
if is_prime(num):
  print(f"{num} is a prime number")
else:
  print(f"{num} is not a prime number")
```

10. Palindrome Check

Python
```
def is_palindrome(string, start, end):
  """
  This function recursively checks if a string is a palindrome.
  """
  if start >= end:
    return True
  elif string[start] != string[end]:
    return False
  else:
    return is_palindrome(string, start + 1, end - 1)

# Example usage:
text = "racecar"
if is_palindrome(text, 0, len(text) - 1):
  print(f"'{text}' is a palindrome")
else:
  print(f"'{text}' is not a palindrome")
```