105. Convert Binary Number to Decimal

The `binary_to_decimal` function takes a binary string as input, splits it into individual digits, reverses them, and then calculates the decimal value by considering each digit's position and value, without using parseInt.

```
def binary_to_decimal(binary)
  binary.reverse.chars.map.with_index { |bit, index|
bit.to_i * 2 ** index }.sum
end

puts binary_to_decimal("1101")
# Output: 13
```

106. Check if an Array is Sorted in Descending Order

The `sorted_descending?` function checks if an array is sorted in descending order.

```ruby
def sorted_descending?(arr)
  arr.each_cons(2).all? { |a, b| a >= b }
end

puts sorted_descending?([5, 4, 3, 2, 1])
# Output: true
puts sorted_descending?([1, 5, 3, 8, 2])
# Output: false
```

107. Find the Average of Even Numbers in an Array

The average_of_even_numbers function finds the average of even numbers in an array.

```
def average_of_even_numbers(arr)
  even_numbers = arr.select(&:even?)
  even_numbers.sum.to_f / even_numbers.length
end

puts average_of_even_numbers([1, 2, 3, 4, 5, 6, 7, 8, 9,
10])
# Output: 6.0
```

108. Capitalize the First Letter of Each Word in a String

The `capitalize_words` capitalizes the first letter of each word in a String.

```ruby
def capitalize_words(str)
  str.gsub(/\b\w/) { |char| char.upcase }
end

puts capitalize_words("hello world")
# Output: "Hello World"
```

109. Check if an Array is a Subset of Another Array

The `is_subset` function takes two arrays arr1 and arr2 as input, and it checks whether every element of arr1 is present in arr2 using the all? method combined with include?. It returns true if all elements of arr1 are found in arr2, and false otherwise.

```
def is_subset(arr1, arr2)
  arr1.all? { |item| arr2.include?(item) }
end

puts is_subset([1, 2, 3], [2, 3, 4, 5, 6])
# Output: false
puts is_subset([1, 2, 3], [2, 3, 1, 5, 6])
# Output: true
```

110. Find the Minimum and Maximum Numbers in an Array

The `min_max` function takes an array arr as input and returns a hash containing the minimum and maximum values of the array using the min and max methods provided by Ruby arrays.

```ruby
def min_max(arr)
  { min: arr.min, max: arr.max }
end

puts min_max([10, 5, 25, 3, 15])
# Output: { min: 3, max: 25 }
```

111. Check if a Number is a Narcissistic Number

This `is_narcissistic_number` takes a number num as input and returns true if it is a narcissistic number and false otherwise.

```
def is_narcissistic_number(num)
  num == num.to_s.chars.map(&:to_i).sum { |digit| digit **
num.digits.length }
end

puts is_narcissistic_number(153)
# Output: true
puts is_narcissistic_number(370)
# Output: true
puts is_narcissistic_number(123)
# Output: false
```

112. Remove Null and Undefined Values from an Array

The `remove_nil_and_undefined` function removes null and undefined values from an array.

```
def remove_nil_and_undefined(arr)
  arr.compact
end

puts remove_nil_and_undefined([1, nil, 2, 3, nil, 4, nil, undefined])
# Output: [1, 2, 3, 4]
```

113. Reverse the Order of Words in a String

The `reverse_words` function reverse the order of words in a String.

```ruby
def reverse_words(str)
  str.split.reverse.join(' ')
end

puts reverse_words("Hello, world!")
# Output: "world! Hello,"
```

114. Calculate the Sum of Cubes of an Array

The `sum_of_cubes` function calculates the sum of cubes of an array.

```ruby
def sum_of_cubes(arr)
  arr.reduce(0) { |acc, val| acc + val ** 3 }
end

puts sum_of_cubes([1, 2, 3, 4, 5])
# Output: 225
```

115. Shuffle the Characters of a String

The `shuffle_string` function suffle the characters of a String.

```ruby
def shuffle_string(str)
  str.chars.shuffle.join
end

puts shuffle_string("hello")
# Output: Randomly shuffled string, e.g., "olelh"
```

116. Find the Nth Fibonacci Number (recursive)

The `fibonacci` function finds the nth fibonacci number (recursive).

```ruby
def fibonacci(n)
  return n if n <= 1
  fibonacci(n - 1) + fibonacci(n - 2)
end

puts fibonacci(7)
# Output: 13
```

117. Count the Words in a String

The `count_words` function splits the input string by whitespace characters using a regular expression (`/\s+/`) and returns the length of the resulting array, which corresponds to the number of words in the string.

```
def count_words(str)
  str.split(/\s+/).length
end

puts count_words("Hello, how are you doing?")
# Output: 5
```

118. Check if a Number is a Triangular Number

The `is_triangular_number` function calculates triangular numbers until the sum exceeds the input number. It then checks if the sum equals the input number and returns true if they are equal, indicating that the input number is a triangular number, and false otherwise.

```
def is_triangular_number(num)
  n = 0
  sum = 0
  while sum < num
    n += 1
    sum += n
  end
  sum == num
end

puts is_triangular_number(10)
# Output: true
puts is_triangular_number(15)
# Output: true
puts is_triangular_number(7)
# Output: false
```

119. Calculate the Perimeter of a Rectangle

The `rectangle_perimeter` function calculates the perimeter of a rectangle given its width and height.

```ruby
def rectangle_perimeter(width, height)
  2 * (width + height)
end

puts rectangle_perimeter(5, 10)
# Output: 30
```

120. Find the Longest Common Prefix in an Array of Strings

The `longest_common_prefix` function finds the longest common prefix among an array of strings.

```
def longest_common_prefix(strs)
  return "" if strs.empty?
  prefix = strs.first
  strs[1..-1].each do |str|
    prefix = prefix.chop until str.start_with?(prefix)
  end
  prefix
end

puts longest_common_prefix(["apple", "apricot",
"appetizer"])
# Output: "app"
```

121. Get the ASCII Value of a Character

The `get_ascii_value` function gets the ASCII value of a character. The ord method is used to get the ASCII value of a character.

```
def get_ascii_value(char)
  char.ord
end

puts get_ascii_value('A')
# Output: 65
```

122. Find the First Non-Repeated Character in a String

The `first_non_repeated_char` function finds the first non-repeated character in a String.

```
def first_non_repeated_char(str)
  char_count = Hash.new(0)
  str.each_char { |char| char_count[char] += 1 }
  str.each_char { |char| return char if char_count[char] ==
1 }
  nil
end

puts first_non_repeated_char('abacabad')
# Output: "c"
```

123. Sort an Array of Objects by a Property Value

The `sort_by_property` function sorts an array of objects by a property value.

```
def sort_by_property(arr, prop)
  arr.sort_by { |obj| obj[prop] }
end

people = [
  { name: "Alice", age: 25 },
  { name: "Bob", age: 20 },
  { name: "Charlie", age: 30 },
]

puts sort_by_property(people, :age)
# Output: [{:name=>"Bob", :age=>20}, {:name=>"Alice",
:age=>25}, {:name=>"Charlie", :age=>30}]
```

124. Calculate the Exponential of a Number

The `exponential` function calculates the result of raising a given base to a specified exponent using the exponentiation operator (**).

```
def exponential(base, exponent)
  base ** exponent
end

puts exponential(2, 3)
# Output: 8
```

125. Check if a String is an Anagram of Another String

The `is_anagram?` function checks if a string is an anagram of another string.

```
def is_anagram?(str1, str2)
  str1.chars.sort.join == str2.chars.sort.join
end

puts is_anagram?("listen", "silent")
# Output: true

puts is_anagram?("hello", "world")
# Output: false
```

126. Find the Factors of a Number

The `factors` function calculates and returns an array of all the factors of a given number. Factors are the positive integers that evenly divide the input number.

```
def factors(num)
  (1..num).select { |i| num % i == 0 }
end

puts factors(12)
# Output: [1, 2, 3, 4, 6, 12]
```

127. Check if a Number is a Neon Number

The `is_neon_number` function checks if a number is a neon number.

```
def is_neon_number(num)
  (num**2).digits.sum == num
end

puts is_neon_number(9)
# Output: true

puts is_neon_number(12)
# Output: false
```

128. Find the Power Set of a Set

The `power_set` function finds the power set of a set.

```
def power_set(set)
  set.reduce([[]]) { |result, item|
result.concat(result.map { |subset| subset + [item] }) }
end

puts power_set([1, 2, 3])
# Output: [[], [1], [2], [1, 2], [3], [1, 3], [2, 3], [1,
2, 3]]
```

129. Check if a Number is a Disarium Number

The `is_disarium_number` function checks if a number is a disarium number by first converting it into an array of its digits, then calculating the sum of powers of its digits, and finally comparing the sum with the original number.

```
def is_disarium_number(num)
  num.to_s.chars.each_with_index.sum { |digit, index|
digit.to_i**(index + 1) } == num
end

puts is_disarium_number(89)
# Output: true

puts is_disarium_number(135)
# Output: true

puts is_disarium_number(23)
# Output: false
```

130. Remove Vowels from a String

The `remove_vowels` takes a string str as input and returns a new string with all vowels removed. It uses the gsub method to replace all occurrences of vowels (both lowercase and uppercase) in the input string with an empty string "". The regular expression /[aeiouAEIOU]/ matches any vowel character (both lowercase and uppercase) within the string.

```ruby
def remove_vowels(str)
  str.gsub(/[aeiouAEIOU]/, "")
end

# Example usage:
puts remove_vowels("Hello, World!")
# Output: "Hll, Wrld!"
```

131. Generate an Array of Consecutive Numbers

The `consecutive_numbers` function generates an array of consecutive numbers.

```
def consecutive_numbers(start, end_num)
  (start..end_num).to_a
end

# Example usage:
puts consecutive_numbers(1, 5).inspect
# Output: [1, 2, 3, 4, 5]
```

132. Check if a Number is a Pronic Number

The `is_pronic_number` function checks if a number is a pronic number.

```ruby
# Check if a Number is a Pronic Number
# The is_pronic_number function determines whether a given number is a pronic number.
def is_pronic_number(num)
  n = Math.sqrt(num).to_i
  n * (n + 1) == num
end

# Example usage:
puts is_pronic_number(6)
# Output: true

puts is_pronic_number(20)
# Output: true

puts is_pronic_number(7)
# Output: false
```

133. Check if a String is a Pangram

The `is_pangram` function checks if a string is a pangram.

```ruby
# Check if a String is a Pangram
# The is_pangram function checks whether a given string is
a pangram.
def is_pangram(str)
  letters = str.downcase.scan(/[a-z]/).uniq
  letters.size == 26
end

# Example usage:
puts is_pangram("The quick brown fox jumps over the lazy
dog")
# Output: true

puts is_pangram("Hello, World!")
# Output: false
```

134. Reverse the Order of Words in a Sentence

The `reverse_sentence` function reverses the order of words in a sentence.

```ruby
def reverse_sentence(sentence)
  sentence.split(" ").reverse.join(" ")
end

# Example usage:
puts reverse_sentence("Hello, how are you doing?")
# Output: "doing? you are how Hello,"
```

135. Calculate the Hypotenuse of a Right-Angled Triangle

The `calculate_hypotenuse` function calculates the hypotenuse of a right-angled triangle.

```
def calculate_hypotenuse(a, b)
  Math.sqrt(a ** 2 + b ** 2)
end

# Example usage:
puts calculate_hypotenuse(3, 4)
# Output: 5.0
```