1. Recursion

Example 1: Max Couple at Inverted Indexes

Recursion1

```python
def max_couple(list, length):
    return max_couple_helper(list, length) # Call the helper function

def max_couple_helper(list, length, max = 0, start = 0): # Helper with 2 new variables
    if length == 0 or (length == 1 and list[length + start - 1] <= max): # Stop condition case1
        return max

    if length == 1 and list[length + start - 1] > max: # Stop condition case2
        return list[length + start - 1]

    if list[start] + list[length + start - 1] > max: # Check if this pair is the new max value
        max = list[start] + list[length + start - 1]

    return max_couple_helper(list, length - 2, max, start + 1)
```

Example 2: Twin Neighbors

Recursion2_Sum

```python
def twin_neighbours(my_list):
    return twin_helper(my_list, len(my_list) - 1)

# Helper for q1
def twin_helper(my_list, last_index, count=0):
    if last_index == 0: # Stop Condition - reached all items in the list
        return 0
    if my_list[last_index] == my_list[last_index - 1]: # When they are twins
        return 1 + twin_helper(my_list, last_index - 1) # Add to sum and continue
    else: # When they're NOT twins
        return twin_helper(my_list, last_index - 1)
```

Example 3: יעני פיבונצ'י

Recursion3_Series

```python
def like_fibo(nth):
    if nth <= 3: # Stop condition - reached the default [1, 2, 3] list
        return nth

    if nth % 2 == 0: # If index is even, return sum of prev 3 items
        return like_fibo(nth - 1) + like_fibo(nth - 2) + like_fibo(nth - 3)
    else: # Index is odd
        return abs(like_fibo(nth - 1) - like_fibo(nth - 3))
```

Example 4: מספר מתחלף

Recursion4

```python
def is_switched_number(number):
    if number < 10: # Stop condition
       return True
    dig0 = number % 10 # Right dig
    dig1 = number // 10 % 10 # Left dig
    if (dig0 % 2 == 0 and dig1 % 2 == 0) or (dig0 % 2 != 0 and dig1 % 2 != 0): # Check condition
       return False

    return is_switched_number(number // 10) # Cut last digit (it passed the test here)
```

2. String

Example 1: Check Email Address Validity

String1

```python
def is_valid_email(address):
    username = address.split('@')[0] # Extract the username
    domain_name = address.split('@')[-1] # Extract the domain name
    country_code = domain_name.split('.') # Extrace the country code

    check1_at_symbol = address.count('@') == 1 # Check if there's 1 '@' symbol
    check2_length = len(address) >= 8 and len(address) <= 30 # Check length validity
    check3_first_char = (address[0]).isalpha() # Check that first char is a letter
    check4_lower_complexity = False # Complexity - lowercase. False by default
    check4_upper_complexity = False # Complexity - uppercase. False by default
    check5_server_validity = '.' in domain_name and len(country_code[-1]) >= 2 # Serv validity
    check6_country_code_validity = True # Country code validity. True by default

    for letter in username: # Check complexity validity
        if letter.islower():
            check4_lower_complexity = True
        elif letter.isupper():
            check4_upper_complexity = True
    for letter in country_code[-1][-2:]: # Check country code validity
        if not letter.isalpha():
            check6_country_code_validity = False

    is_valid = (check1_at_symbol and check2_length and check3_first_char and check4_lower_complexity \
                and check4_upper_complexity and check5_server_validity and check6_country_code_validity)
    return is_valid
```

Example 2: Capitalize Words

String2

```python
def capitalize_words(input_string):
    str_as_list = input_string.split(' ') # Split the words into a list
    str_as_list = [word.capitalize() for word in str_as_list if word != ''] # Capitalize each word and remove empty words

    res_str = ' '.join(str_as_list) # Convert from list to string
    return res_str
```

3. Lists

Example 1: Rotate Matrix

נקציה המקבלת מטריצה ומסובבת אותה ב- 90 מעלות עם כיוון השעון (ימינה). אין להשתמש ברשימות עזר. יש לבצע הכל על המטריצה המקורית וללא slicing.

Lists1

```python
def rotate_matrix_90_degrees_clockwise_v1(matrix):
    for i in range(len(matrix) // 2): # Run on first half of rows
        for j in range(len(matrix) // 2): # Run on first half of columns
        # together it runs on the frames from outside -> inside
            top_left = matrix[i][j] # Top left item
            top_right = matrix[j][-i - 1] # Top right item
            bottom_right = matrix[-i - 1][-j - 1] # Bottom right item
            bottom_left = matrix[-j - 1][i] # Bottom left item

            temp = top_left # Save top left into a temp, to perform the rotation

            matrix[i][j] = bottom_left # Bot left into Top left
            matrix[-j - 1][i] = bottom_right # Bot right into Bot left
            matrix[-i - 1][-j - 1] = top_right # Top right into Bot right
            matrix[j][-i - 1] = temp # Top left into Top right

    return matrix
```

Example 2: Create Snake

Lists2

```python
arr = [[None] * cols for i in range(rows)] # Create an empty null 2d arr
    value = 1 # Starting value. Will be bumped by 1 for each index

    for col in range (-1, -(len(arr[0])) - 1, -1): # Run on columns from right to left
        for row in range(len(arr)): # Run on rows
            if col % 2 != 0: # Check if row is even or odd
                arr[-row - 1][col] = value
            else:
                arr[row][col] = value
            value += 1 # Bump value by 1. Now ready to insert into next index

    return arr
```

Example 3: Diagram Graph

Lists3

```python
def diagram_graph(list):
    max_val = max(list) # Find the max value (highest graph bar)

    for row in range(max_val, 0, -1): # Run on rows from top to bottom
        for bar in range(len(list)): # Run on bars (columns)
            if list[bar] >= row: # Print '*' if bar reaches this row (height)
                print('*', end = ' ')
            else: # Skip/Print ' ' if bar doesn't reach this row (height)
                print(' ', end = ' ')
        print() # Proceed to next row

    #Cosmetic:
        # Add cosmetic lines here
```

Example 4: חיבור ארוך

Lists4

```python
def sum_lists (list1, list2):
    longer_l, shorter_l = (list1, list2) if len(list1) >= len(list2) else (list2, list1)
    summed_list = [None] * (len(longer_l) + 1) # Define an empty list with +1 size
    leftover = 0 # Leftover will be 1 when sum of 2 nums is greater than 10

    for i in range(len(longer_l)): # Run on the longer list of the two
        item = longer_l[-i -1] + leftover # Get the items from right to left
        if i < (len(shorter_l)): # ONLY if this index exists in the shorter list
            item += shorter_l[-i -1] # Add the corresponding item from shorter lst
        if item > 9: # Define leftover if sum > 9
            item -= 10
            leftover = 1
        else:
            leftover = 0
        summed_list[-i -1] = item # Add the sum to result list

    if leftover == 1:
        summed_list[0] = 1
    else:
        summed_list.remove(None)

    return summed_list
```

4. Dictionary

Dictionary

```python
def get_books_name_for_reader(books, readers, reader_name): # Exercise 2's Function
    leased_books = []
    for reader in readers:
        if reader['name'] == reader_name: # Found the dictionary of our reader
            for book_id in reader['borrowed']:
                for book in books:
                    if book_id == book['book_id']:
                        leased_books.append(book['title'])
    return leased_books


def most_read_book(books, readers): # Exercise 3's Function
    most_leased = set()
    highest_votes = 0
    for book in books:
        book['votes'] = 0
        for reader in readers:
            for book_id in reader['borrowed']:
                if book_id == book['book_id']:
                    book['votes'] += 1
        highest_votes = book['votes'] if book['votes'] > highest_votes else highest_votes
    for book in books:
        if book['votes'] == highest_votes:
            most_leased.add(book['title'])
    return most_leased


def readers_having_most_read_book(readers): # Exercise 6's Function
    books_by_id = []
    possessing_readers = set()
    highest_votes = 0
    for reader in readers: # Create a list with dicts containing book ID book's vote count
        for readers_book_id in reader['borrowed']:
            books_by_id.append(dict(book_id = readers_book_id, votes = 0))

    for book in books_by_id: # Count the votes and find the highest vote count
        for reader in readers:
            for readers_book_id in reader['borrowed']:
                if readers_book_id == book['book_id']:
                    book['votes'] += 1
        highest_votes = book['votes'] if book['votes'] > highest_votes else highest_votes

    for book in books_by_id: # Check which readers have the most-leased books
        if book['votes'] == highest_votes:
            for reader in readers:
                for readers_book_id in reader['borrowed']:
                    if readers_book_id == book['book_id']:
                        possessing_readers.add(reader['name'])

    return possessing_readers


if __name__ == '__main__':
    # Books List
    books = [dict(book_id=1001, title="Harry Potter", genre="fantasy", pages=500),
             dict(book_id=1002, title="A song of Ice and Fire", genre="fantasy", pages=700),
             dict(book_id=1003, title="1984", genre="classic", pages=800),
             dict(book_id=1004, title="Attack on Titan", genre="manga", pages=1400),
             dict(book_id=1005, title="One Piece", genre="manga", pages=12000) ]
    # Readers List
    readers = [{"name": "Ichi", "borrowed": [1001, 1003]},
               {"name": "Ni", "borrowed": [1002]},
               {"name": "San", "borrowed": [1005, 1002]},
               {"name": "Yon", "borrowed": [1005, 1002]},
               {"name": "Go", "borrowed": [1005]} ]
```