```python
# Title: Programming Assignment 1 Part A
# Due date: Wednesday, September 1, 2021 at 11:59pm
# Author: Sotheanith Sok
# Description: Practice working with matrices

# -------------------------------------------------------------------------------
# imports
import numpy as np

# Given
A = np.array([[1, -2, 4, 5], [3, -1, 9, -7], [8, 5, 4, 0], [0, -3, 2, 1]])
B = np.array([[3, 1, 2, 7], [4, 6, 5, 0], [-1, 3, 2, 5], [-6, -13, 0, -2]])
print("Given:\nA:\n%s\nB:\n%s\n" % (A, B))

# 1. How many rows A has?
num_row_A = A.shape[0]
print("1. How many rows A has?")
print("Ans: %s\n" % num_row_A)

# 2. Show the whole first to third rows of A?
first_to_third_rows_A = A[0:3, :]
print("2. Show the whole first to third rows of A?")
print("Ans:\n%s\n" % first_to_third_rows_A)

# 3. Show the sub-matrix of A starting from second row to the last row, and third column to the fourth one.
sub_matrix_A = A[1:, 2:]
print(
    "3. Show the sub-matrix of A starting from second row to the last row, and third column to the fourth one."
)
print("Ans:\n%s\n" % sub_matrix_A)

# 4. Add 10 to the first row of B, then add the first row to the second row (row1 = 10 + row1, row2 = row1 + row2). Next replace the first row of A with the second row of B.
B[0, :] = 10 + B[0, :]
B[1, :] = B[0, :] + B[1, :]
A[0, :] = B[1, :]
print(
    "4. Add 10 to the first row of B, then add the first row to the second row (row1 = 10 + row1, row2 = row1 + row2). Next replace the first row of A with the second row of B."
)
print("Ans:\nA:\n%s\nB:\n%s\n" % (A, B))

# 5. Find the elements of A less than 5 and greater or equal to -2. What are their indices?
indices = np.where(np.logical_and(A < 5, A >= -2))
indices = list(zip(indices[0], indices[1]))
print(
    "5. Find the elements of A less than 5 and greater or equal to -2. What are their indices?"
)
print("Ans: %s\n" % indices)

# 6. Find the first 6 indices corresponding to the nonzero entries of A.
indices = np.where(A != 0)
indices = list(zip(indices[0][:6], indices[1][:6]))
print("6. Find the first 6 indices corresponding to the nonzero entries of A.")
print("Ans: %s\n" % indices)

```

```python
# 7. What is the smallest, largest, and average value of A?
smallest = A.min()
largest = A.max()
average = np.average(A)
print("7. What is the smallest, largest, and average value of A?")
print("Ans:\nSmallest: %s\nLargest: %s\nAverage: %s\n" % (smallest, largest, average))

# 8. Write a vector with equally spaced elements from 5 to 0, with a step of 0.3, but in decreasing order. What will be the size?
vec = np.arange(5, 0, -0.3)
print(
    "8. Write a vector with equally spaced elements from 5 to 0, with a step of 0.3, but in decreasing order. What will be the size?"
)
print("Ans: %s\n" % vec.shape[0])

# 9. Create a 3x4 matrix of random numbers between 0 and 1
matrix = np.random.rand(3, 4)
print("9. Create a 3x4 matrix of random numbers between 0 and 1")
print("Ans:\n%s\n" % matrix)
```

```python
# Title: Programming Assignment 1 Part B
# Due date: Wednesday, September 1, 2021 at 11:59pm
# Author: Sotheanith Sok
# Description: Implement binary search function

# ------------------------------------------------------------------------------
# imports
import numpy as np

def binary_search_recursive(array, key, start, end):
    # Not found case
    if start > end:
        return -1

    # Searching...
    else:
        # Find mid index
        mid = (start + end) // 2

        # Check if value at mid index is the key
        if array[mid] == key:
            return mid

        # If value at mid smaller than key, search right side...
        elif array[mid] < key:
            return binary_search_recursive(array, key, mid + 1, end)

        # If value at mid index larger than key, search left side...
        else:
            return binary_search_recursive(array, key, start, mid - 1)


done = False

while not done:
    # Get array length
    while True:
        n = input("Enter array length (enter \"done\" to exit): ")
        if n == "done":
            done = True
            break
        elif (n.isnumeric() and int(n) > 0):  # isnnumeric guarantee 0 and positive integers
            n = int(n)  # Convert input to int
            break

    # Check if program should exit
    if done:
        break

    # Generate, sort, and print array
    a = np.random.randint(-10, 11, n)
    a = np.sort(a)
    print("a = %s" % a)

    # Get key
    while True:
        key = input("Enter key (enter \"done\" to exit): ")
        if key == "done":
            done = True
            break
        elif (key.isnumeric() and int(key) > 0):  # isnnumeric guarantee 0 and positive integers
            key = int(key)  # Convert input to int
            break

    # Check if program should exit
    if done:
        break

    # Call binary search algorithm on the array and the key
    result = binary_search_recursive(a, key, 0, n - 1)

    # Print result
    if result == -1:
        print('Result: Key not found!!!')
    else:
        print('Result: Key found at index %s.' % result)
```