Computer Organization and Structure

Please write the following four programs in MIPS assembly language.

1. **Josephus Problem (30%)**
   There are N people standing in a circle waiting to be executed. The counting out begins at some point in the circle and proceeds around the circle in a fixed direction. In each step, a certain number of people are skipped and the next person is executed (every K-th person is executed). The elimination proceeds around the circle (which is becoming smaller and smaller as the executed people are removed), until only the last person remains, who is given freedom.

   In the following, n denotes the number of people in the initial circle, and k denotes the count for each step, that is, k-1 people are skipped and the k-th is executed. The people in the circle are numbered from 1 to n .

   We will give you two positive numbers. They are both greater than 1. One denotes the number of people in the initial circle and the other denotes the count for each step. Your job is to write a program to determine the last survivor number.

   Your output should look like this. The filename is **Josephus.s**.

   ```
   ———— Josephus Problem ————
   Input :
   n = 5
   k = 2

   Output :
   Last survivor = 3
   ```

2. **Greatest Common Divisor (35%)**
   We will give you two positive integers. They are both greater than 1. Your job is to calculate their greatest common divisor.

   Your output should look like this. The filename is **GCD.s**.

   ```
   ———— Greatest Common Divisor ————
   Input :
   x = 45
   y = 18

   Output :
   gcd(x, y) = 9
   ```

3. **Tower of Hanoi (35%)**
   Your task is to complete a recursive implementation of the Tower of Hanoi problem in order to get familiar with assembly programming. We will give you an integer n ($1 \leq n \leq 5$) which means Tower of Hanoi with n disks. Your job is to move all disks from A to C and print out the moving process.

   Your output should look like this. The filename is **Hanoi.s**.

   ```
   ---- Tower of Hanoi ----
   Input :
   n = 3

   Output :
   move a -> c
   move a -> b
   move c -> b
   move a -> c
   move b -> a
   move b -> c
   move a -> c
   ```

4. **Bonus: Sorting Algorithm (optional)**
   We will give you five positive integers. Your job is to sort 5 numbers in ascending order. You can choose a sorting algorithm you want. (Such as Quick sort, Merge sort, Bubble sort …)

   Your output should look like this. The filename is **Sorting.s**

   ```
   ---- Sorting Algorithm ----
   Input :
   Insert the first integer: 3
   Insert the second integer: 4
   Insert the third integer: 2
   Insert the fourth integer: 5
   Insert the fifth integer: 1

   Output :
   1, 2, 3, 4, 5
   ```