1. Write a recursive procedure\(^1\) to compute and return the greatest common divisor (GCD) of 2 integer arguments. First write the procedure in a HLL and then compile the HLL code to MIPS assembly. The GCD can be defined recurrently as:
   a. \( \text{GCD}(a, b) = b \) if \( b \) divides \( a \)
   b. \( \text{GCD}(a, b) = \text{GCD}(b, r) \) otherwise, where \( r = a \mod b \)

2. Write a procedure called `precision()` with no arguments. The procedure must return the smallest floating point number that can be added to 1 such that the result of the sum is different from 1.

3. Write a procedure `roots(a,b,c)` that receives the three float coefficients of a polynomial and returns an integer representing the number of distinct real roots.

4. Write a procedure `sin(x)` that takes one float argument representing an angle in radians. The procedure should return the approximated floating point value of `sin(x)` by computing the sum of a Taylor series.

5. Write procedures that take a one dimensional array of integers \( a \) and its length and perform the following operations:
   a. Multiply the array by a scalar
   b. Compute the sum of the elements of the array
   c. Sort the array increasingly

6. Repeat problem 3 this time using arrays of double precision floating point numbers.

7. Write a procedure `mmult(a,b,c,n)` that takes as arguments three square matrices \( a \), \( b \) and \( c \) with common length \( n \). The procedure should compute the matrix product of \( a \) and \( b \) a store the result on matrix \( c \). First write a HLL version of `mmult` and then hand-compile it to MIPS assembly language.

8. All the problems on Chapters 3 and 4 of Patterson and Hennessy *Computer Organization and Design.*

---

\(^1\) In all problems provide answers in both High Level Language and Assembly Language