List of objectives of each programming assignment:

Note that each programing assignment is building upon previous assignments.

1. Program1 (difficulty level: basic): focus on designing basic control flow statements plus integer arithmetic.
    a. if statements (If, Else if, Else).
    b. for loops
    c. while loops
    d. do while loops
    e. basic integer arithmetic, integer division up to a specific decimal points all using integer instructions (no floating point)

2. Program 2 (difficulty level: intermediate): focus on subprograms designs, passing arguments IN and OUT using $a and $v registers, passing base address of static array and basic array functions.
    a. subprograms
    b. static arrays of integers
    c. passing arguments using registers
    d. array methods, such as read_array, print_array, minimum, maximum and average (using integer division)

3. Program 3 (difficulty level: intermediate): more subprogram designs so students feel comfortable using subprograms and practicing modular design, passing base address of dynamic array and understanding the similarities and differences between them (static vs dynamic).
    a. subprograms
    b. dynamic arrays of integers
    c. passing arguments using registers
    d. basic array manipulations, such as concatenation and sum (sigma)

4. Program 4 (difficulty level: intermediate): mastering subprogram concept, introducing use of stack to pass arguments IN and OUT and properly using stack (following all the steps).
    a. subprograms
    b. dynamic arrays of integers
    c. passing arguments using stack
    d. intermediate array manipulations, such as indexOf and reverse

5. Program 5 (difficulty level: intermediate): mastering the use of stack, introducing floating point registers, instructions and conversion between floating points and integers, proper allocation of dynamic array of floating points numbers.

    a. passing arguments using stack
    b. allocation, reading and printing of dynamic array of floating point numbers
    c. floating point arithmetic, conversions and branches
    d. advanced array manipulations, partition_array and remove_by_value

6. Program 6 (difficulty level: advanced): introducing concept of 2-dimensional arrays (row-major and column major) and basic matrix arithmetic.
    a. 2-dimensional array of integers (use column-major for programing assignment)
    b. matrix arithmetic, addition, subtraction and multiplication

7. Program 7 (difficulty level: advanced): focusing on implementing algorithms (algorithm to assembly code), recursive subprograms and reusing concepts such floating points and stack. Implementing functionalities such as:
    a. floating point ← sqrt(integer)
    b. integer ← factorial(integer)
    c. floating point ← round(floating point)
    d. floating point ← floor(floating point) using only round() function
    e. floating point ← ceil(floating point) using only round() function