Computer programming and programming languages – 1st year Civ.Eng., 2nd semester

Proposed lab-work content:

1. Presentation of the laboratory and equipment, safety measures, organisational aspects. How to carry out the activities (grades for homework and activity), use of equipment and peripherals, accessible resources. Logical schemes: symbols, their composition, aspects of structuring and generalisation ("Dijkstra" structural primitives). Exercises with logical schemes: max. (a, b, c); max. (a, b, c, d); max. (x_i, i=1..n).
2. Numerical and logical expressions in Fortran, priority of operators (unary/binary, arithmetic, character concatenation, relational, logical), exercises (arithmetic expressions with intrinsic functions, logical expressions). Discussion of type declaration (INTEGER, REAL, COMPLEX, CHARACTER, LOGICAL, BYTE, TYPE) with KIND and mentioning attributes (PARAMETER, POINTER, TARGET, INTENT, DIMENSION, ALLOCATABLE, EXTERNAL, INTRINSIC, PUBLIC, PRIVATE, OPTIONAL, SAVE). Default rule for numeric variable type. Redefinition of operators (example with INTERFACE OPERATOR). The DATA specification.
3. Presentation of fixed and free format. I/O instructions (READ, ACCEPT, PRINT, WRITE), jump instructions (unconditional, calculated or assigned GOTO), IF (arithmetic, simple logic, structured logic), STOP and PAUSE. Comments (!). Force2 development environment interface. Creating a console application from a logical scheme. Handling of compilation and link editing errors. Tracing and debugging programs. Solve a second order equation (logical scheme + program). Transcribing a logical scheme: max. (a, b, c) – depending on the available time…
4. Detailing DIMENSION, the order in which the elements of an array are stored in memory (indices). Position indices for subarrays. DO, WHILE, CONTINUE, CYCLE and EXIT instructions. Exercises with strings (logical schema + program): extreme values, sorting by pivot and flag (bubble) methods.
5. The SELECT CASE structure. Computing the area and perimeter of a circle with radius R and of a right-angled triangle (with sides A, B, C) with restart options (logic schemes + program).
6. Exercises on two-dimensional arrays: transposition of a matrix, multiplication of a matrix by a scalar, sum of the terms on the diagonal of a matrix (logical schema + program).
7. (From here on, the logical schemes no longer make sense) Introduction to the FORMAT specification and the briefly the usual format descriptors for inputs and outputs (A, B, O, Z, H, F, G, D, E, I, L, ', ") and control (BN, BZ, P, Q, S, SP, SS, T, TL, TR, X, $, \, /, :), examples and exercises. Use of logical units (files), OPEN statement (with mention of parameters: ACCESS=, BLANK=, DISP=, ERR=, FILE/NAME=, KEY=, ORGANISATION=, READONLY, SHARED, STATUS/TYPE=, UNIT=) and CLOSE (with UNIT=, DISP/STATUS=, ERR=). Reading the file specification from the keyboard. UNLOCK, REWIND, REWRITE, ENDFILE commands. Beware of G95 compatibility (does not support everything presented as theory) in Force2.
8. Program units (main program, external procedures, modules, BLOCK DATA), short examples. Declaring (as external or internal procedures) and calling subroutines and functions, passing values (arguments, labels, results - emphasise the difference between a subroutine and a function). Calculating mathematical expressions using subroutines and intrinsic functions, exercises.
9. Exercises with user-defined subroutines and functions (as internal procedures and as external procedures), possibly using intrinsic subroutines and functions. ENTRY points and return variants (RETURN no.) of subroutines. Recursive functions (RECURSIVE, RESULT), example.
10. Introduction to the 3 types of dynamic memory allocation (ALLOCATABLE, POINTER, automatic). Exercises on vectors and 2D arrays, matrix operations (using data input and result files, with filenames read from the keyboard) with dynamic memory allocation (examples: sum of positive elements, number of negative elements, maximum difference between adjacent elements, etc.).
11. Exercises with arrays, through dynamic memory allocation, using user-defined subroutines and functions or format descriptors (e.g. transpose an array, multiply arrays, add arrays, etc.).
    Instead of the variant below, possible with physical presence in the laboratories:
    CVF / Intel Visual Fortran development environment interface. Creating a console application in Developer Studio. Handling compilation and link-editing errors. Tracking (execution tracing) and

debugging programs. Linking a library (IMSL). Showing some IMSL subroutines and how to find them (using the PDF documentation on the lab computers).

12. Continue the matrix exercises using user-defined subroutines and functions or format descriptors in combination with SELECT CASE (e.g. number of positive/negative/null elements, sum of positive/negative/null elements, etc.).
    Instead of the variant below, which would have been possible with physical presence in the laboratories:
    Matrix operations using IMSL subroutines (display variants, multiplication, inversion, etc.).
13. Exercises with pointers (dealing with an input/output string character by character).
14. *Practical test.* Grading and discussing the work during the semester.