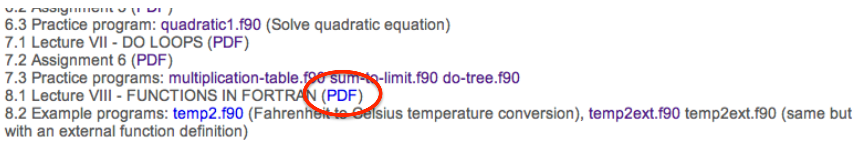
Computers in Chemistry – Lecture VIII

Prof. Dr. Stephan Irle
Quantum Chemistry Group
Nagoya University

Get this lecture online

- Please go to: http://qc.chem.nagoya-u.ac.jp
- Click on "Teaching"
- Click on "PPT" link of "8.1 Lecture VIII – Functions in FORTRAN"

  userid: qcguest, password: qcigf!



6.2 Assignment 5 (PDF)
6.3 Practice program: quadratic1.f90 (Solve quadratic equation)
7.1 Lecture VII - DO LOOPS (PDF)
7.2 Assignment 6 (PDF)
7.3 Practice programs: multiplication-table.f90 sum-to-limit.f90 do-tree.f90
8.1 Lecture VIII - FUNCTIONS IN FORTRAN (PDF)
8.2 Example programs: temp2.f90 (Fahrenheit to Celsius temperature conversion), temp2ext.f90 temp2ext.f90 (same but with an external function definition)

Today's Lecture

- Programming with Functions
- Combine FORTRAN statements into a program unit that can be used in similar or different contexts.
- Saves time, makes code easier to read
- Especially useful for complex problems:

1. Divide the problem into small pieces
2. write functions and/or subroutines to solve them
3. put them together in one complete program.

6.2 Functions I

- FORTRAN language provides many *intrinsic*, or *library,* functions.
- Include numeric functions such as cos(), exp(), abs(), etc. shown in Lecture 5 (Table 2-2), as well as character and logical functions.
- For a complete list, see for example http://www.nsc.liu.se/~boein/f77to90/a5.html
- Today we would like to add new, user-defined functions

6.2 Functions II

- Functions are written as “**function subprograms**”, which are separate program units with similar syntax to that of a FORTRAN “**program**”:

```
Function heading
Specification part
Execution part
END FUNCTION statement
```

- The function can be contained in the same or a different .f90 file. For simplicity, we will only consider the case where one .f90 source code file contains both **program** and **functions**.

6.2 Functions III

- Function heading is a FUNCTION statement of the form:

```
FUNCTION function-name (formal-argument-list)
```

- Or:

```
type-identifier FUNCTION function-name (formal-argument-list)
```

- “function-name” is a legal Fortran identifier, “formal-argument-list” is an identifier or list (possibly empty, in which case we still need “()”) of identifiers separated by commas, and in the second version, “type-identifier” is the name of a FORTRAN type (integer, real, etc.)

6.2 Functions IV

- Variables in the “formal-argument-list” are called “**formal**” or “**dummy arguments**” and are used to pass information to the function subprogram.
- Note: Different program languages have different default ways of passing information from the main program to the subprogram.
- FORTRAN: “pass-by-reference” (use a memory pointer)
- C/C++ and Java: “pass-by-value” (the value cannot be changed by the subprogram)

6.2 Functions V

- Specfication part of a function subprogram has the same form as that of a regular program. It must declare:

1. The type of the function value of not declared in the function heading
2. The type of each formal argument appearing in the “list-of-arguments” as well as variables that appear in the function subprogram

- The execution part of a function subprogram is similar to a regular program, except that it has to include at least one statement:

```
function-name = expression
```

- The last statement of a function subprogram should be:

```
END FUNCTION function-name
```

6.2 Functions VI

- The function value of the function subprogram will be returned to the calling program, when a “RETURN” statement is executed.

```
RETURN
```

- Example: download a program to convert temperature from Fahrenheit to Celsius units, temp2.f90, and compile and run it in an X-Windows terminal by:
- cd Downloads
- gfortran –o temp2.x temp2.f90
- ./temp2.x

6.2 Functions VII

- Sample run:

```
$ ./temp2.x
 Enter temperature in Fahrenheit:
32
   32.00000     is equivalent to   0.000000     in Celsius
 More tmperatures to convert (Y/N)?
y
[stephan@hawk ~]$ ./temp2.x
 Enter temperature in Fahrenheit:
32
   32.00000     is equivalent to   0.000000     in Celsius
 More tmperatures to convert (Y/N)?
Y
 Enter temperature in Fahrenheit:
212
   212.0000     is equivalent to   100.0000     in Celsius
 More tmperatures to convert (Y/N)?
Y
 Enter temperature in Fahrenheit:
-22.5
  -22.50000     is equivalent to  -30.27778     in Celsius
 More tmperatures to convert (Y/N)?
```

6.2 Functions VIII

- In this program, I have used an “internal” function subprogram, which is included BEFORE the END PROGRAM statement.
- Alternative ways: EXTERNAL function subprogram, temp2ext.f90
- Download, run, and compare the two programs
- NOTES: INTENT(IN) protects the variable from being modified within the function subroutine
- EXTERNAL statement is necessary when using an external function subprogram that appears outside the PROGRAM (within the same .f90 file or within different .f90 files)

6.2 FUNCTIONS IX

- Task: Write a FORTRAN 90 program containing a real-valued function NumericGrade that accepts a letter grade and returns the corresponding numeric value (A = 4.0, B = 3.0, C = 2.0, D = 1.0, F = 0.0)

Good luck. This concludes today’s lecture.