032CM - 2026

PROGRAMMING FOR COMPUTATIONAL CHEMISTRY

Fortran: Control constructs (loops, if condition)

Gianluca Levi

gianluca.levi@units.it, giale@hi.is

Office: Building C11, 3rd floor, Room 329

Fall 2026

Do loop

Iterative / counting loop, used to **repeat a block of instructions** a fixed number of times

```
do i = istart, iend, incr
  statements
enddo
```

i → loop counter (integer)
istart → starting value
iend → final value
incr → increment (optional, default 1)

The loop counter cannot be modified inside the loop

Nested loops are allowed

Do loop

Iterative / counting loop, used to **repeat a block of instructions** a fixed number of times

```
do i = istart, iend, incr
  statements
enddo
```

i → loop counter (integer)
istart → starting value
iend → final value
incr → increment (optional, default 1)

Try it!

Write a Fortran program that computes y = 0.1 + 0.1 + … (**10 times**) using a Do loop.

Do while loop

Repeat a block of instructions **while a logical condition is true**

```
do while (condition)
  statements
end do
```

The condition is checked before each iteration

The loop terminates as soon as the condition becomes **false**

Useful when the **number of iterations is not known in advance**

Try it!

Write a Fortran program that computes y = 0.1 + 0.1 + … (**10 times**) using a Do while loop.

If condition

Execute a block of statements **only if a condition is true**

```
if (condition) then
  statements
endif
```

Multiple options (**IF – ELSEIF – ELSE**):

```
if (condition1) then
  statements1
elseif (condition2) then
  statements2
else
  statements3
endif
```

Short form (single statement):

```
if (condition) instruction
```

Assignment 2

Problem 2

Many fractions cannot be represented exactly in binary floating-point arithmetic. This exercise explores how the types and precision of the operands affect numerical results. Assume that default `real` arithmetic uses single precision (32 bits) and `double precision` arithmetic uses 64 bits.

(a) Write a Fortran program that evaluates the following three expressions exactly as written. Store the three sums and the reference value in part (b) in separate variables declared as `double precision`.

(i) `1.d0/3.  + 1/3`

(ii) `1./3.  + 1.d0/3.`

(iii) `1.d0/3.  + 1./3.d0`

Print each sum with 15 digits after the decimal point.

Hint: If a sum is stored in a variable named `result`, you can print it using

```
write(*,'(F25.15)') result
```

The format `F25.15` uses a field 25 characters wide with 15 digits after the decimal point.

(b) Compute a reference value for $\frac{2}{3}$ using double-precision division. Store it in the reference variable and print it using the same format as in part (a).

(c) Compare each sum from part (a) with the reference value from part (b). State which stored results are exactly equal to the reference, and explain any differences by considering the types of the operands and the precision used in each operation.

Assignment 2

Problem 3

The factorial of a positive integer n is defined as

$$n! = n \times (n-1) \times (n-2) \times \cdots \times 2 \times 1,$$

with the special case

$$0! = 1.$$

Write a Fortran program that asks the user to input an integer n. If $n > 0$, the program should compute $n!$ using a loop. If $n = 0$, the program should return the result $0! = 1$. If $n < 0$, the program should raise an error message and stop execution. Print the result with a clear message, e.g. `Factorial of n is ....`