Fortran 90 Cheat Sheet

by karlp via cheatography.com/167060/cs/34965/

Boiler plate program

```
program name
        imp licit none
       ! module imports go here
       ! local variables go here
       ! code goes here, there
is not main entry point.
end program
```

Module

```
module name
        imp licit none
       use other_ module !
import another module
       ! local variables go here
        con tains
       ! functions and subrou -
tines goes here
end module
```

Function

```
! return type goes before the
function
integer function f()
       f = 42 ! the function
name is result variable
end function
```

Subroutine

```
! subroutines does not return a
value
! but they have out parameters
subroutine sub(a, b, c)
        int eger, intent(in) ::
a
        int eger, intent(in) ::
b
        int eger, intent (out)
:: c
       c = a + b
end subroutine
```

Loops

```
do while (logical expr)
     ! Control:
     ! exit for break
     ! cycle for continue
end do
label: do i = start, stop ! ,
step ! (optional)
       ! statements
end do
```

Conditions

```
if (cond) then
       ! ...
else if (cond) then
       ! ...
else
       ! ...
end if
```

Operators

+ -	addition, substraction
* /	multiplication, division
**	exponentiation
==	equality (numbers)
/=	inequality (numbers)
>, <	greater/less than
>=, <=	greater/less or equal
.eqv.	equality (booleans)
.neqv.	inequality (booleans)

Operators (cont)

.and., .or., .not.	logical and, or and not.

Native functions

abs(v)	absolute value of v
aimag(z)	imaginary part of z (single prec)
int(v, kind)	truncates toward zero to convert to integer
ceiling(v, kind)	ceiling and convert
floor(v, kind)	floor and convert
modulo(a, p)	polymorphic modulo (sign of p)
cmplx(x, y, kind)	make a complex from floats
conjg(z)	conjugate complex
cos(x), dcos(x), ccos(x)	cosine
sin(x), dsin(x), csin(x)	sine
acos(x), dacos(x)	inverse cosine
asin(x), dasin(x)	inverse sine
dprod(x, y)	x * y as a double
exp(x), dexp(x), cexp(x)	exponential
erf(x), derf(x)	error function
erfc(x), derfc(x)	complementary error function

Native functions (cont)

hypot(a, b)	hypothenuse of x and y (single prec)
log(x), log10(x)	natural and base 10 logarithm
max(a, b, ...), min(a, b, ...)	polymorphic extrema
sum(arr), product(arr), minval(arr), maxval(arr), all(arr), any(arr)	polymorphic reduction of array
sum(arr, dim), product(arr, dim), minval(arr, dim), maxval(arr, dim), all(arr, dim), any(arr, dim)	polymorphic reduction of array along axis dim
minloc(arr), maxloc(arr)	the coordi-anates of an extrema as a 1D array
minloc(arr, dim), maxloc(arr, dim)	the indices of extrema in along axis dim

Data types

`logical`	boolean
`intege r(k ind=1)`	8 bits signed integer
`intege r(k ind=4)`	32 bits signed integer
`intege r(k ind=8)`	64 bits signed integer
`real(k ind=4)`	simple precision float
`real(k ind=8)`	double precision float
`comple x(k ind=4)`	simple precision complex number

Data types (cont)

`comple x(k ind=8)`	double precision complex number
`charac ter (len=n)`	string of size n
`integer, dimens ion(m, n)`	2D array of shape (m, n)

Derived types

```
type :: name
    integer :: i
   real :: x
   ! more fields
end type
type(s ome _type) :: obj
! Initialize
obj%fi eld_a = 1
obj%fi eld_b = 0.5
```