Tecniche di programmazione in chimica computazionale

Derived data types, I/O format & files

Emanuele Coccia

Dipartimento di Scienze Chimiche e Farmaceutiche

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name

- Component selector %

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name

- Component selector %
- Read and write statements for the full derived-type variable

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name
- Component selector %
- Read and write statements for the full derived-type variable
- Easy to treat with single components of the derived-type variable (as ordinary variables)

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name
- Component selector %
- Read and write statements for the full derived-type variable
- Easy to treat with single components of the derived-type variable (as ordinary variables)
- Used as dummy arguments in procedures

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name
- Component selector %
- Read and write statements for the full derived-type variable
- Easy to treat with single components of the derived-type variable (as ordinary variables)
- Used as dummy arguments in procedures
- Example example1.f90

Derived type

- User-defined combination of existing types

  type name

  ...

  components

  ...

  end type name
- Component selector %
- Read and write statements for the full derived-type variable
- Easy to treat with single components of the derived-type variable (as ordinary variables)
- Used as dummy arguments in procedures
- Example example1.f90
- Example example2.f90

Read and write statements

- General structure:

  read(UNIT=u,FMT=label) input data

Read and write statements

- General structure:

  read(UNIT=u,FMT=label) input data

  - *UNIT=u*: input channel (* or 5 is standard input)

Read and write statements

- General structure:

  read(UNIT=u,FMT=label) input data

  - *UNIT=u*: input channel (* or 5 is standard input)
  - *FMT=label*: format of input data (* free format)

Read and write statements

- General structure:

  read(UNIT=u,FMT=label) input data

  - *UNIT=u*: input channel (* or 5 is standard input)
  - *FMT=label*: format of input data (* free format)

- General structure:

  write(UNIT=u,FMT=label) output data

Read and write statements

- General structure:

  read(UNIT=u,FMT=label) input data

  - *UNIT=u*: input channel (* or 5 is standard input)
  - *FMT=label*: format of input data (* free format)

- General structure:

  write(UNIT=u,FMT=label) output data

  - *UNIT=u*: output channel (* or 6 is standard input)

Read and write statements

- General structure:

  read(UNIT=u,FMT=label) input data

  - *UNIT=u*: input channel (* or 5 is standard input)
  - *FMT=label*: format of input data (* free format)

- General structure:

  write(UNIT=u,FMT=label) output data

  - *UNIT=u*: output channel (* or 6 is standard input)
  - *FMT=label*: format of output data (* free format)

Format statement

- Used by read or write statements

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:
  - lw: integer number with a field of w characters

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:
  - Iw: integer number with a field of w characters
  - Fw.d: real number with a field of w characters (sign and dot included), d decimal places

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:
  - Iw: integer number with a field of w characters
  - Fw.d: real number with a field of w characters (sign and dot included), d decimal places
  - Ew.d: real number with a field of w characters (sign and dot included), d decimal places, exponent ($w \geq d + 7$)

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:
  - Iw: integer number with a field of w characters
  - Fw.d: real number with a field of w characters (sign and dot included), d decimal places
  - Ew.d: real number with a field of w characters (sign and dot included), d decimal places, exponent ($w \geq d + 7$)
  - nX: n blank spaces

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:
  - Iw: integer number with a field of w characters
  - Fw.d: real number with a field of w characters (sign and dot included), d decimal places
  - Ew.d: real number with a field of w characters (sign and dot included), d decimal places, exponent ($w \geq d + 7$)
  - nX: n blank spaces
  - Aw: character type variable in a field of w characters

Format statement

- Used by read or write statements
- In any point of the program:

  label FORMAT(descriptors list)

- Descriptors list:
  - Iw: integer number with a field of w characters
  - Fw.d: real number with a field of w characters (sign and dot included), d decimal places
  - Ew.d: real number with a field of w characters (sign and dot included), d decimal places, exponent ($w \geq d + 7$)
  - nX: n blank spaces
  - Aw: character type variable in a field of w characters
- Example format.f90

File management

- File: many lines of data

File management

- File: many lines of data
- Data accessed as a unit

File management

- File: many lines of data
- Data accessed as a unit
- Each line of information corresponds to a record

File management

- File: many lines of data
- Data accessed as a unit
- Each line of information corresponds to a record
- Fortran reads/writes information one record at a time

File management

- File: many lines of data
- Data accessed as a unit
- Each line of information corresponds to a record
- Fortran reads/writes information one record at a time
- Read from/write to a file is a slow operation: sequential or direct access
- Formatted or unformatted data

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name
  - *STATUS='st'*: old, new, scratch, unknown (default), replace (for output)

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name
  - *STATUS='st'*: old, new, scratch, unknown (default), replace (for output)
  - *ACCESS='acc'*: sequential (default) or direct

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name
  - *STATUS='st'*: old, new, scratch, unknown (default), replace (for output)
  - *ACCESS='acc'*: sequential (default) or direct
  - *ACTION='act'*: read, write or readwrite (default)

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name
  - *STATUS='st'*: old, new, scratch, unknown (default), replace (for output)
  - *ACCESS='acc'*: sequential (default) or direct
  - *ACTION='act'*: read, write or readwrite (default)
  - *FORM='fm'*: formatted (default if sequential) or unformatted (default if direct)

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name
  - *STATUS='st'*: old, new, scratch, unknown (default), replace (for output)
  - *ACCESS='acc'*: sequential (default) or direct
  - *ACTION='act'*: read, write or readwrite (default)
  - *FORM='fm'*: formatted (default if sequential) or unformatted (default if direct)
  - *IOSTAT=ios*: different from 0 if an error occurs

File management

- To connect an external file to an I/O unit:

  OPEN(UNIT=u,FILE='name',STATUS='st',ACCESS='acc',
  ACTION='act',FORM='fm',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *FILE='name'*: file name
  - *STATUS='st'*: old, new, scratch, unknown (default), replace (for output)
  - *ACCESS='acc'*: sequential (default) or direct
  - *ACTION='act'*: read, write or readwrite (default)
  - *FORM='fm'*: formatted (default if sequential) or unformatted (default if direct)
  - *IOSTAT=ios*: different from 0 if an error occurs
  - *ERR=label*: if an error occurs, go to the instruction labeled with *label*

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *STATUS='st'*: keep (default) or delete

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *STATUS='st'*: keep (default) or delete
  - *IOSTAT=ios*: different from 0 if an error occurs

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *STATUS='st'*: keep (default) or delete
  - *IOSTAT=ios*: different from 0 if an error occurs
  - *ERR=label*: if an error occurs, go to the instruction labeled with *label*

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *STATUS='st'*: keep (default) or delete
  - *IOSTAT=ios*: different from 0 if an error occurs
  - *ERR=label*: if an error occurs, go to the instruction labeled with *label*
- BACKSPACE: move back one record in a file

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *STATUS='st'*: keep (default) or delete
  - *IOSTAT=ios*: different from 0 if an error occurs
  - *ERR=label*: if an error occurs, go to the instruction labeled with *label*
- BACKSPACE: move back one record in a file
- REWIND: move to the beginning of a file

File management

- To disconnect an external file from an I/O unit:

  CLOSE(UNIT=u,STATUS='st',IOSTAT=ios,ERR=label)

  - *UNIT=u*: mandatory, indicates I/O unit associated to the file
  - *STATUS='st'*: keep (default) or delete
  - *IOSTAT=ios*: different from 0 if an error occurs
  - *ERR=label*: if an error occurs, go to the instruction labeled with *label*
- BACKSPACE: move back one record in a file
- REWIND: move to the beginning of a file
- Examples file1.f90, file2.f90, file3.f90 and file4.f90