fll – Fortran Linked List Library Introduction – v1.1

Adam Jirasek

Introduction

- Available at gihub.com/libm3l/fll
- LGPL OSS license

- Multi level, doubly linked list
- Fortran language
- Most of functions similar names to Unix/Linux

Introduction

- List consists of nodes
    - type of directory “DIR” or “N”
    - Type of file (R,D,I,L,S etc….)
        - R – real number
        - D - double real number
        - I – integer
        - L – long integer
        - S – fixed length string

Introduction

- Example

```
Main_List DIR 3
   Subdir DIR 2
      pressure D 5  1
         1 2 3 4 5
       density D 1 5
          3 4 5 6 7
    Subdir   DIR 1
        volumes D 5 1
          1.5 2.5 3.5 4.5 5.5
    Index   L 5 2
       3 5 7 9 10 4 5 6 7 8
```

Introduction

- Above list starts with MainDir which contains three data sets
  - Two subdirectories
  - And one data set

  - The first subdirectory contains two data sets
    - Pressure, type double, array is 1D and, length is five and contains values 1 2 3 4 5
    - Density, type double, 1D array, length 5, contains 3 4 5 6 7
    - NOTE: both arrays are 1D and will be stored in 1D array even though the index of the second suggest the array has 5 columns
  - The second subdirectory contains one data set
    - Volumes, 1D array, type long integer, length 5, contains values 1.5 2.5 3.5 4.5 5.5
  - The third data se it a 2D array of long integers

Introduction

- Available functions
  - **fll_mv** - move node
  - **fll_cp** - copy node
  - **fll_mklist** - make node
  - **fll_locate** - locate node
  - **fll_nnodes** – get number of nodes
  - **fll_getndata** – get data of node
  - **fll_rm** - remove node
  - **fll_cat** - print node
  - **fll_read** - read list from a file
  - **fll_write** - write list to a file
  - **fll_read_ffa** - read list from FFA format file
  - **fll_write_ffa** - write list to FFA format file
  - **fll_deattach** – detaches node from list

  - Each function or subroutine has fpar

Function fll_cp()

- **fll_cp(pwhat, pwhere, fpar)**
  - Copies pwhat node to pwhere
    - If pwhere = NULL(), the function duplicates pwhat node
  - Return value – pointer to a new copy

Function fll_mv()

- **fll_mv(pwhat, pwhere, fpar)**
  - Moves pwhat node to pwhere
  - Return value - logical value, return value can be true or false depending on if the move operation was successful

Function fll_mk()

- **fll_mk(name,type,ndim,nsize,fpar)**
  - Makes a new node of list
  - Input – name of node, type of node, first and second dimensions
    - If type of node is DIR, ndim and nsize are automatically set to 0
  - Return - pointer to newly created node

Function fll_locate()

- **fll_locate (pnode,name,number,type,dim,recursive,fpar)**
  - Locates node
  - Input parameters
    - Pnode – list where to search
    - Name – name of node
    - Number – order of the node (1st, 2nd etc…) if more nodes of the same name
    - Type – type of node
    - Dim – dimensions of arrays in the node, can be 0,1,2, if any other number the dimensions is not considered
    - Recursive – search list recursively, if so, number == 1
    - Both name and type can be set to *
  - Return – pointer to located node

Function fll_nnodes()

- **fll_locate (pnode,name,number,type,dim,recursive,fpar)**
  - Return number of nodes pnode list
  - Input parameters
    - Pnode – list where to search
    - Name – name of node
    - Number – order of the node (1st, 2nd etc…) if more nodes of the same name
    - Type – type of node
    - Dim – dimensions of arrays in the node, can be 0,1,2, if any other number the dimensions is not considered
    - Recursive – search list recursively, if so, number == 1
    - Both name and type can be set to *
  - Return – number of nodes

Function fll_getndata()

- **fll_getndata(pnode,name,number,type,recursive,fpar)**
  - Returns data in nodes which are not type of DIR
  - Input parameters
    - Pnode – list where to search
    - Name – name of node
    - Number – order of the node (1st, 2nd etc…) if more nodes of the same name
    - Type – type of node
    - Dim – dimensions of arrays in the node, can be 0,1,2, if any other number the dimensions is not considered
    - Recursive – search list recursively, if so, number == 1
    - Both name and type can be set to *
  - Return – pointer to the data

Function fll_getndata()

- Functions are
  - Real numbers
    - **fll_getndata_r0**
    - **fll_getndata_r1**
    - **fll_getndata_r2**
  - Double numbers
    - **fll_getndata_d0**
    - **fll_getndata_d1**
    - **fll_getndata_d2**
  - Strings
    - **fll_getndata_s0**
    - **fll_getndata_s1**
    - **fll_getndata_s2**

Subroutine fll_rm()

- **fll_getndata(pnode,fpar)**
  - Removes data
  - Input parameters
    - Pnode – list to be removed
  - Return – pointer to the data

Subroutine fll_cat()

- **fll_getndata(pnode,iounit,parent,fpar)**
  - Prints data to iounit
  - Input parameters
    - Pnode – list to be printed
    - Iounit – number of file descriptor
    - Parent – if TRUE write information about node's parent

Subroutine fll_cat()

- **fll_deattach(pnode,fpar)**
  - Detaches PNODE from list
    - After being detached from list, the node parent and siblings are NULL
    - The node is removed from the list
    - The function is an opposite to fll_mv() function
  - Input parameters
    - Pnode – list to be printed
    - Parent – if TRUE write information about node’s parent

Subroutine fll_write()

- **fll_write(pnode,file,iounit,fmt,fpar)**
  - Write data to FLL native format file
  - Input parameters
    - Pnode – list to be printed
    - File – name of file
    - Iounit - number of file descriptor
    - Fmt – A- asci file, B – binary file

Subroutine fll_read()

- **fll_read(pnode,file,iounit,fmt,fpar)**
  - Read data from FLL native format file
  - Input parameters
    - Pnode – list to be printed
    - File – name of file
    - Iounit – number of file descriptor
    - Fmt – A- asci file, B – binary file

Subroutine fll_write()

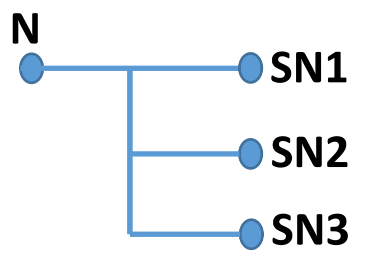
- **fll_write_ffa(pnode,file,iounit,fmt,fpar)**
  - Write data to FFA format file
  - Input parameters
    - Pnode – list to be printed
    - File – name of file
    - Iounit - number of file descriptor
    - Fmt – A- asci file, B – binary file

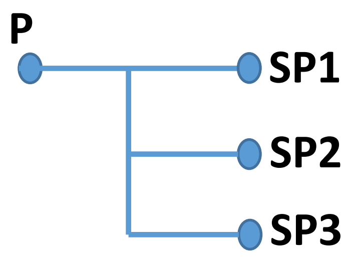
Subroutine fll_read()

- **fll_read_ffa(pnode,file,iounit,fmt,fpar)**
  - Read data from FFA format file
  - Input parameters
    - Pnode – list to be printed
    - File – name of file
    - Iounit – number of file descriptor
    - Fmt – A- asci file, B – binary file

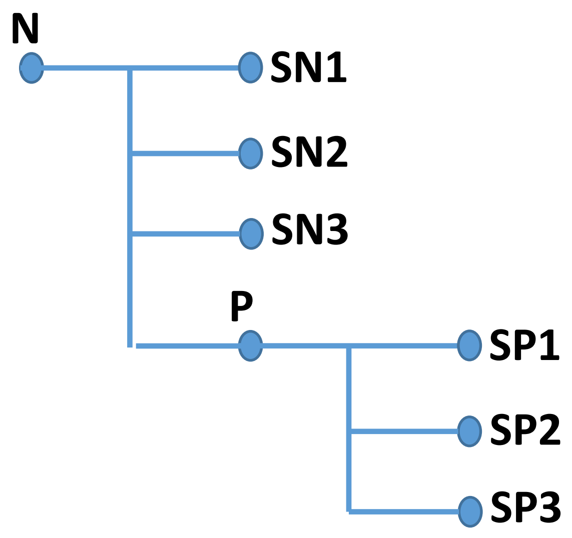
Moving, copying nodes details

- N node is a DIR type of node, SN1, SN2, SN3 are data type of nodses

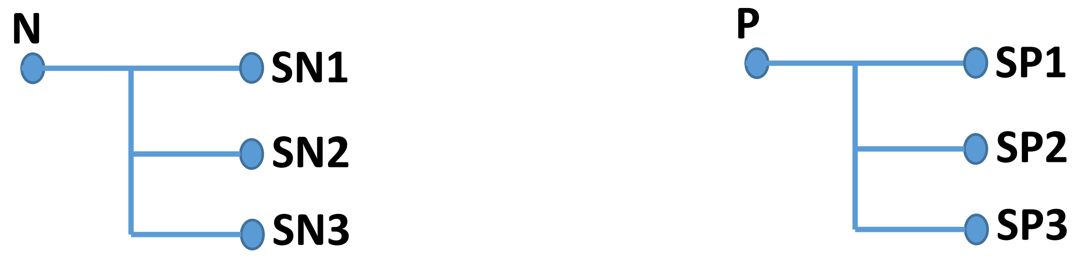




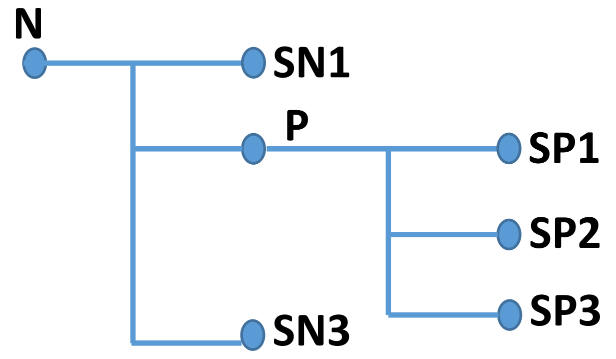
1. **fll_mv(P,N,fpar)** will result in node P being moved into node N as a new subset



Moving, copying nodes details



1. **fll_mv(P,SN2,fpar)** will result in node SN2 being overwritten by node P, original node SN2 and its data will be removed


N
SN1
P
SP1
SP2
SP3
SN3