```
-- file TreeDefs.Mesa
-- last modified by Satterthwaite, May 4, 1978  11:48 AM

DIRECTORY
  TableDefs: FROM "tabledefs",
  LitDefs: FROM "litdefs",
  SymDefs: FROM "symdefs";

TreeDefs: DEFINITIONS =
  BEGIN

  treetype: TableDefs.TableSelector = TableDefs.chunktype;

 -- data structures

  TreeLink: TYPE = RECORD [
    SELECT tag: * FROM
      subtree => [index: TreeIndex],
      hash => [index: SymDefs.HTIndex],
      symbol => [index: SymDefs.ISEIndex],
      literal => [info: LitDefs.LitRecord],
      ENDCASE];

  TreeId: TYPE = RECORD [baseP: TableDefs.TableFinger, link: TreeLink];


  TreeNode: TYPE = MACHINE DEPENDENT RECORD [
    free: BOOLEAN,        -- reserved for allocator
    name: NodeName,
    mark: BOOLEAN,
    shared: BOOLEAN,
    attr1, attr2: BOOLEAN,
    nsons: INTEGER [0..MaxNSons],
    info: UNSPECIFIED,
    son1: TreeLink,
    son2: TreeLink,
    son3: TreeLink,
    son4: TreeLink,
    son5: TreeLink,
    son6: TreeLink];     -- and as many more as needed

  MaxNSons: CARDINAL = 7;

  TreeNodeSize: CARDINAL = 2;    -- minimum node size

  TreeIndex: TYPE = POINTER [0..TableDefs.TableLimit) TO TreeNode;

  nullTreeIndex: TreeIndex = FIRST[TreeIndex];
  empty: TreeLink = TreeLink[subtree[index: nullTreeIndex]];

  nullid: TreeLink = TreeLink[hash[index: SymDefs.HTNull]];


  NodeName: TYPE = {
   -- general tree constructors
    list, item,

   -- declarations
    declitem,
    modeTC, basicTC, enumeratedTC, recordTC, monitoredTC, variantTC,
    pointerTC, arrayTC, arraydescTC,
    procTC, processTC, portTC, signalTC, errorTC, programTC,
    definitionTC, unionTC, relativeTC,
    subrangeTC, longTC,
    implicitTC, frameTC, discrimTC,
    entry, --internal,--
    unit, --diritem,-- module, body, inline, block, lambda,

  -- statements
    assign, construct, vconstruct, rowcons, extract,
    ifstmt,
    casestmt, casetest, caseswitch,
    bindstmt,
    dostmt, forseq, upthru, downthru,
    return,
    goto, exit, loop,
    syserror,
    resume, continue, retry, catchmark,
    restart, stop,
    wait, notify, broadcast, unlock,
    nullstmt,
    label,
    openstmt,
    enable, catchphrase,
    dst, lst, lstf,
    procinit,

   -- expressions
    apply, call, portcall, signal, error, xerror, start, fork, join,
    index, dindex,  seqindex, reloc,
    constructx, vconstructx, unionx, rowconsx,
    ifexp, caseexp, bindexp,
    assignx,
    or,
    and,
    relE, relN, relL, relGE, relG, relLE, in, notin,
    plus, minus,
    times, div, mod,
    dot, cdot, dollar,
    not,
    uminus,
    addr,
    uparrow,
    min, max, lengthen, abs,
    size, first, last,
    arraydesc,
    length, base,
    loophole,
    register, memory,
    intOO, intOC, intCO, intCC,
    clit, llit,
    cast, float, align,
    openexp,
    new,
    stringinit,
    signalinit, portinit,
    mwconst,
    temp,

    none,

  -- late additions (should be moved)
    internal, diritem
    };

 -- tree construction interface

  TreeInit, TreeErase: PROCEDURE;

  mlpush: PROCEDURE [v: TreeLink];
  mlpop: PROCEDURE RETURNS [TreeLink];

  mlinsert: PROCEDURE [TreeLink, CARDINAL];
  mlextract: PROCEDURE [CARDINAL] RETURNS [TreeLink];

  maketree: PROCEDURE [name: NodeName, count: INTEGER] RETURNS [TreeLink];
  makelist: PROCEDURE [size: INTEGER] RETURNS [TreeLink];

  pushtree: PROCEDURE [name: NodeName, count: INTEGER];
  pushlist, pushproperlist: PROCEDURE [size: INTEGER];
  pushhashtree: PROCEDURE [hti: SymDefs.HTIndex];
  pushsymtree: PROCEDURE [sei: SymDefs.ISEIndex];
  pushlittree: PROCEDURE [lti: LitDefs.LTIndex];
  pushstringlittree: PROCEDURE [sti: LitDefs.STIndex];

  setinfo: PROCEDURE [info: UNSPECIFIED];
  setattr: PROCEDURE [attr: [1..2], value: BOOLEAN];

 -- tree deallocation

  freenode: PROCEDURE [node: TreeIndex];
  freetree: PROCEDURE [t: TreeLink] RETURNS [TreeLink];
```

```
-- tree attributes

 GetNode: PROCEDURE [t: TreeLink] RETURNS [TreeIndex];
 shared: PROCEDURE [t: TreeLink] RETURNS [BOOLEAN];
 setshared: PROCEDURE [t: TreeLink, shared: BOOLEAN];
 testtree: PROCEDURE [t: TreeLink, name: NodeName] RETURNS [BOOLEAN];

-- tree manipulation

 TreeScan: TYPE = PROCEDURE [t: TreeLink];
 TreeMap: TYPE = PROCEDURE [t: TreeLink] RETURNS [v: TreeLink];

 UpdateTree: PROCEDURE [root: TreeLink, map: TreeMap] RETURNS [v: TreeLink];

-- list testing

 listlength: PROCEDURE [t: TreeLink] RETURNS [CARDINAL];
 listhead: PROCEDURE [t: TreeLink] RETURNS [TreeLink];
 listtail: PROCEDURE [t: TreeLink] RETURNS [TreeLink];

-- list manipulation

 scanlist: PROCEDURE [root: TreeLink, action: TreeScan];
 reversescanlist: PROCEDURE [root: TreeLink, action: TreeScan];
 updatelist: PROCEDURE [root: TreeLink, map: TreeMap] RETURNS [TreeLink];
 reverseupdatelist: PROCEDURE [root: TreeLink, map: TreeMap] RETURNS [TreeLink];


-- cross-table tree copying

 CopyTree: PROCEDURE [root: TreeId, map: TreeMap] RETURNS [v: TreeLink];
 IdentityMap: TreeMap;
 NodeSize: PROCEDURE [baseP: TableDefs.TableFinger, node: TreeIndex] RETURNS [CARDINAL];

 END.
```