```
-- FileLookup.Mesa; edited by Sandman on October 17, 1977  8:29 AM

DIRECTORY
  AltoFileDefs: FROM "altofiledefs",
  DirectoryDefs: FROM "directorydefs",
  FileLookupDefs: FROM "filelookupdefs",
  InlineDefs: FROM "inlinedefs",
  IODefs: FROM "iodefs",
  SystemDefs: FROM "systemdefs",
  StringDefs: FROM "stringdefs",
  SegmentDefs: FROM "segmentdefs";

DEFINITIONS FROM SegmentDefs;

FileLookup: PROGRAM IMPORTS DirectoryDefs, IODefs, SegmentDefs, StringDefs, SystemDefs
  EXPORTS FileLookupDefs, SegmentDefs SHARES SegmentDefs =
BEGIN

FP: TYPE = AltoFileDefs.FP;

Entry: TYPE = RECORD [
  link: POINTER TO Entry,
  name: STRING,
  fp: FP];

HashSize: CARDINAL = 19;
HashIndex: TYPE = [0..HashSize);

HashVector: ARRAY HashIndex OF POINTER TO Entry;

HashValue: PROCEDURE [s: STRING] RETURNS [HashIndex] =
  BEGIN OPEN InlineDefs;
  i: CARDINAL;
  i ← BITAND[LOOPHOLE[s[0]],137B] + BITAND[LOOPHOLE[s[s.length/2]],137B];
  RETURN[BITXOR[i,s.length*17B] MOD HashSize];
  END;

InsertEntry: PROCEDURE [name: STRING, fp: POINTER TO FP] =
  BEGIN
  hv: HashIndex = HashValue[name];
  entry: POINTER TO Entry;
  entry ← SystemDefs.AllocateHeapNode[SIZE[Entry]];
  entry.name ← name;
  entry.fp ← fp↑;
  entry.link ← HashVector[hv];
  HashVector[hv] ← entry;
  END;

FindEntry: PROCEDURE [name: STRING, fp: POINTER TO FP] RETURNS [BOOLEAN] =
  BEGIN
  hv: HashIndex = HashValue[name];
  entry: POINTER TO Entry;
  FOR entry ← HashVector[hv], entry.link UNTIL entry = NIL DO
    IF StringDefs.EquivalentString[name,entry.name] THEN
      BEGIN
      fp↑ ← entry.fp;
      RETURN[TRUE];
      END;
    ENDLOOP;
  RETURN[FALSE];
  END;

GetFileName: PUBLIC PROCEDURE [file: FileHandle] RETURNS [STRING] =
  BEGIN
  hv: HashIndex;
  entry: POINTER TO Entry;
  localname: STRING ← [40];
  heapname: STRING;
  FOR hv IN HashIndex DO
    FOR entry ← HashVector[hv], entry.link UNTIL entry = NIL DO
      IF entry.fp.serial = file.fp.serial THEN RETURN[entry.name];
      ENDLOOP;
    ENDLOOP;
  IF ~DirectoryDefs.DirectoryLookupFP[@file.fp,localname] THEN
    SIGNAL InvalidFP[@file.fp];
  heapname ← SystemDefs.AllocateHeapString[localname.length];
  StringDefs.AppendString[heapname,localname];
  InsertEntry[heapname,@file.fp];
  RETURN[heapname];
  END;

NewFile: PUBLIC PROCEDURE [
  name:STRING, access:AccessOptions, version:VersionOptions]
  RETURNS [FileHandle] =
  BEGIN OPEN InlineDefs;
  fp: FP;  old, create: BOOLEAN;
  [access,version] ← ValidateOptions[access,version];
  create ← BITAND[version,OldFileOnly]=0;
  old ← FindEntry[name,@fp];
  IF ~old THEN old ← DirectoryDefs.DirectoryLookup[@fp,name,create];
  IF (old AND BITAND[version,NewFileOnly]#0)
  OR (~old AND ~create) THEN ERROR FileNameError[name];
  RETURN[InsertFile[@fp,access]];
  END;

ValidateOptions: PROCEDURE [
  access:AccessOptions, version:VersionOptions]
  RETURNS [AccessOptions, VersionOptions] =
  BEGIN OPEN InlineDefs;
  IF access = DefaultAccess THEN access ← Read;
  -- IF version = DefaultVersion THEN version ← 0;
  IF BITAND[version,NewFileOnly+OldFileOnly] = NewFileOnly+OldFileOnly
  OR (BITAND[version,NewFileOnly]#0 AND BITAND[access,Append]=0)
    THEN ERROR FileAccessError[NIL];
  IF BITAND[access,Append]=0 THEN
    version ← BITOR[version,OldFileOnly];
  RETURN[access,version]
  END;

-- FileRequests

FileRequest: TYPE = RECORD [
  link: POINTER TO FileRequest,
  name: STRING];

RequestHead: POINTER TO FileRequest ← NIL;

AddFileRequest: PUBLIC PROCEDURE [name: STRING] =
  BEGIN
  r: POINTER TO FileRequest = SystemDefs.AllocateHeapNode[SIZE[FileRequest]];
  r.name ← name;
  r.link ← RequestHead;
  RequestHead ← r;
  END;

FilesMissing: PUBLIC ERROR = CODE;

ProcessFileRequests: PUBLIC PROCEDURE =
  BEGIN
  r, next: POINTER TO FileRequest;
  checkone: PROCEDURE [fp: POINTER TO AltoFileDefs.FP, name: STRING] RETURNS [BOOLEAN] =
    BEGIN
    r, next: POINTER TO FileRequest;
    prev: POINTER TO FileRequest ← NIL;
    FOR r ← RequestHead, next UNTIL r = NIL DO
      next ← r.link;
      IF StringDefs.EquivalentString[r.name,name] THEN
        BEGIN
        InsertEntry[r.name, fp];
        IF prev = NIL THEN RequestHead ← next
        ELSE prev.link ← next;
        SystemDefs.FreeHeapNode[r];
        EXIT;
        END;
      prev ← r;
      ENDLOOP;
    RETURN[RequestHead = NIL]
    END;

  DirectoryDefs.EnumerateDirectory[checkone];
  IF RequestHead # NIL THEN
    BEGIN OPEN IODefs;
    WriteLine["Files not found:"];
    FOR r ← RequestHead, next UNTIL r = NIL DO
      next ← r.link;
      WriteChar[' ]; WriteLine[r.name];
      SystemDefs.FreeHeapNode[r];
      ENDLOOP;
    RequestHead ← NIL;
    ERROR FilesMissing;
    END;
  END;

-- Main body
i: HashIndex;
FOR i IN HashIndex DO HashVector[i] ← NIL ENDLOOP;

END...
```