```
(****
 *
 * Object FileSpace and its subsidiary objects define an abstract file space
 * used in the file management operations below.
 *
 *)

module File;

  from Rolodex import Rolodex, Card, Name;

  obj FileSpace is File*;
  obj File is type:FileType and n:FileName and fd:FileData;
  obj FileType is at:ASCIIType or bt:BinaryType or rt:RolodexType;
  obj FileName is Name;
  obj FileData is ad:ASCIIData or bd:BinaryData or rd:RolodexData;
  obj ASCIIData is string*;
  obj BinaryData is number*;
  obj RolodexData is Rolodex;
  obj ASCIIType;
  obj BinaryType;
  obj RolodexType;


  operation New()->r':Rolodex
    description: (*
        Create a new empty rolodex.
    *);
    post: r' = nil;
  end New;

  operation Open(fs:FileSpace, n:Name)->r':Rolodex
    description: (*
        Open an existing rolodex file of the given name.  Create a rolodex
        containing the data of the opened file.
    *);
    post:
        (*
         * There is a file of the given name in either ASCII or Rolodex format
         * in the given file space.  If the latter, then the cards of the
         * output rolodex contain the converted ASCII data.  If the former
         * (i.e., file is in Rolodex format), then the cards of the output are
         * those in the file.
         *)
        exists (f in fs)
            (* File and rolodex names are the same *)
            (r'.n = f.n)

                and

            (* File data are the same as rolodex cards, parsed if ASCII *)
            if (f.fd?ad) then
                r'.cs = ParseRolodexText(f.fd.ad)
            else if (f.fd?rd) then
                r' = f.fd.rd
            else false; (* Neither ascii nor Rolodex file format *)

  description: (*
      Open is essentially a find operation, with the same underlying form of
      postcondition as a basic find.  This more concrete version of Open must
      parse ASCII text data files, if that is format in which a rolodex was
      saved.

      The postcondition relies on a non-trivial auxiliary function,
      ParseRolodexText, that would specify precisely the correspondence
      between the textual and abstract representations of rolodex card data.
  *);
end Open;

operation Save(fs:FileSpace, r:Rolodex)->fs':FileSpace
  description: (*
      Save the given rolodex in the given filesapce.

      Save is essentially an add operation.  It should be foramally specified
      in a comparable manner.
  *);
end Save;

operation SaveAs(fs: FileSpace, r:Rolodex, n:Name)->fs':FileSpace
  description: (*
      Save the given rolodex in the given filesapce under the given name.
  *);
  post: ;  (* Comparable change to Save. *)

end SaveAs;

operation Print(r:Rolodex)->rp:RolodexPrintout
  description: (*
      Print the given Rolodex in textual format.
  *);
  post:
      exists (r':Rolodex, c:Card) (
          ((c in r'.cs) iff (c in r.cs)) and
          SortedByName(r'.cs) and
          (rp = GeneratePrint(r'))
      );

  description: (*
      Printing produces the ASCII text for a sorted Rolodex.  The
      postcondition here relies on a non-trivial auxiliary function,
      GeneratePrintText, that would define textual formatting details
      precisely.  GeneratePrintText is the complement of the earlier
      ParseRolodexText function.

      Note that the postcondition specifies sorting on the rolodex, rather
      than on the printout, as would be the case with the following logic:

          forall (c in r)
              (GenerateCardPrint(c) in rp) and SortedByName(rp)

      This logic for sorting a printout would be considerably more tedious,
      since it would have to perform parsing to extract the name field from
```

```
        the text representation of a card.
    *);

  end Print;

  operation Quit is
    inputs: ;
    outputs: ;
    description: (*
        Quit the rolodex system.
    *);
  end Quit;

  obj RolodexPrintout;

  (**
   * The following auxiliary functions should be defined as discussed in the
   * formal spec primer
   *)
  function ParseRolodexText(ASCIIData)->Card*;
  function GeneratePrint(Rolodex)->RolodexPrintout;
  function SortedByName(Card*)->boolean;

end File;
```