1 ÏÕÖ×package body Discrete\_Set is  
 2 ÏÏ§  
 3 ÏÏ§----------------------------------------------------------------------------

-- function Is\_Member determines if “Element” is in the given set, it will return

-- true or false   
 4 ÏÏ§ÏÞßàfunction Is\_Member (Set : in Set\_Type;   
 5 ÏÏ§ÏÏ§Ï Element : in Element\_Type) return Boolean is   
 6 ÏÏ§ÏÏ§begin  
 7 ÏÏ§Â¹Ä¹¹Ïreturn Set(Element);  
 8 ÏÏ§ÏÏ©end Is\_Member;  
 9 ÏÏ§  
10 ÏÏ§----------------------------------------------------------------------------

-- function + returns the union of the two sets given as parameters, basially

-- combining the two sets as one  
11 ÏÏ§ÏÞßàfunction "+" (Left : in Set\_Type; Right : in Set\_Type) return Set\_Type is  
12 ÏÏ§ÏÏ§begin  
13 ÏÏ§Â¹Ä¹¹Ïreturn Left or Right;  
14 ÏÏ§ÏÏ©end "+";  
15 ÏÏ§  
16 ÏÏ§----------------------------------------------------------------------------

-- function + returns a set containing with a new element added in that was given

-- as a parameter  
17 ÏÏ§ÏÞßàfunction "+" (Left : in Set\_Type;   
18 ÏÏ§ÏÏ§Ï Right : in Element\_Type) return Set\_Type is  
19 ÏÏ§ÏÏ§ÏíÏResult : Set\_Type;  
20 ÏÏ§ÏÏ§begin  
21 ÏÏ§ÏÏ¨¹¹ÏResult := Left;  
22 ÏÏ§ÏÏ¨¹¹ÏResult(Right) := True; -- Add the new element to the set  
23 ÏÏ§Â¹Ä¹¹Ïreturn Result;  
24 ÏÏ§ÏÏ©end "+";  
25 ÏÏ§  
26 ÏÏ§----------------------------------------------------------------------------

-- function + returns a set containing with a new element added in that was given

-- as a parameter  
27 ÏÏ§ÏÞßàfunction "+" (Left : in Element\_Type;  
28 ÏÏ§ÏÏ§Ï Right : in Set\_Type) return Set\_Type is  
29 ÏÏ§ÏÏ§ÏíÏResult : Set\_Type;  
30 ÏÏ§ÏÏ§begin  
31 ÏÏ§ÏÏ¨¹¹ÏResult := Right;  
32 ÏÏ§ÏÏ¨¹¹ÏResult(Left) := True; -- Add the new element to the set  
33 ÏÏ§Â¹Ä¹¹Ïreturn Result;  
34 ÏÏ§ÏÏ©end "+";  
35 ÏÏ§  
36 ÏÏ§----------------------------------------------------------------------------

-- function \* returns the intersection of the two sets, this is the elements in each

-- set that are equivalent and returns this set  
37 ÏÏ§ÏÞßàfunction "\*" (Left : in Set\_Type; Right : in Set\_Type) return Set\_Type is  
38 ÏÏ§ÏÏ§begin  
39 ÏÏ§Â¹Ä¹¹Ïreturn Left and Right;  
40 ÏÏ§ÏÏ©end "\*";  
41 ÏÏ§  
42 ÏÏ§----------------------------------------------------------------------------

-- function – returns the difference of the two sets

-- this the set that includes of of left minus the elements of right  
43 ÏÏ§ÏÞßàfunction "-" (Left : in Set\_Type; Right : in Set\_Type) return Set\_Type is  
44 ÏÏ§ÏÏ§begin  
45 ÏÏ§Â¹Ä¹¹Ïreturn Left and not Right;  
46 ÏÏ§ÏÏ©end "-";  
47 ÏÏ§  
48 ÏÏ§----------------------------------------------------------------------------  
-- function – removes the given element from the set and then returns the set

49 ÏÏ§ÏÞßàfunction "-" (Left : in Set\_Type;  
50 ÏÏ§ÏÏ§Ï Right : in Element\_Type) return Set\_Type is  
51 ÏÏ§ÏÏ§ÏíÏResult : Set\_Type;  
52 ÏÏ§ÏÏ§begin  
53 ÏÏ§ÏÏ¨¹¹ÏResult := Left;  
54 ÏÏ§ÏÏ¨¹¹ÏResult(Right) := False; -- Remove the element from the set  
55 ÏÏ§Â¹Ä¹¹Ïreturn Result;  
56 ÏÏ§ÏÏ©end "-";  
57 ÏÏ§  
58 ÏÏ§----------------------------------------------------------------------------

-- function <= decides if is all the members of set left also belong to set right

-- returns true or false, depending

59 ÏÏ§ÏÞßàfunction "<=" (Left : in Set\_Type; Right : in Set\_Type) return Boolean is  
60 ÏÏ§ÏÏ§ÏíÏIs\_A\_Subset : Boolean;  
61 ÏÏ§ÏÏ§begin  
62 ÏÏ§ÏÏ¨¹¹ÏIs\_A\_Subset := (Left - Right) = Empty\_Set;  
63 ÏÏ§Â¹Ä¹¹Ïreturn Is\_A\_Subset;  
64 ÏÏ§ÏÏ©end "<=";  
65 ÏÏ§  
66 ÏÏ§----------------------------------------------------------------------------

-- function <= decides if is all the members of set left also belong to set right

-- returns true or false, depending  
67 ÏÏ§ÏÞßàfunction "<" (Left : in Set\_Type; Right : in Set\_Type) return Boolean is  
68 ÏÏ§ÏÏ§ÏíÏResult : Boolean;  
69 ÏÏ§ÏÏ§begin  
70 ÏÏ§ÏÏ¨¹³´if Left = Right then -- If the sets are equal, not a proper subset  
71 ÏÏ§ÏÏ§Ï6¾¹¹ÏResult := False;  
72 ÏÏ§ÏÏ§Ïö´else  
73 ÏÏ§ÏÏ§Ï¸¾¹¹ÏResult := Left <= Right; -- If not equal, test for subset  
74 ÏÏ§ÏÏ§ÏÈÏend if; -- using the function above  
75 ÏÏ§Â¹Ä¹¹Ïreturn Result;  
76 ÏÏ§ÏÏ©end "<";  
77 ÏÏ§  
78 ÏÏ§----------------------------------------------------------------------------

-- function >= decides if all the members of set right belong to set left

-- returns true or false, depending   
79 ÏÏ§ÏÞßàfunction ">=" (Left : in Set\_Type; Right : in Set\_Type) return Boolean is  
80 ÏÏ§ÏÏ§begin  
81 ÏÏ§Â¹Ä¹¹Ïreturn Right <= Left; -- Reverse the order of the   
82 ÏÏ§ÏÏ§ -- parameters and use the operation   
83 ÏÏ§ÏÏ§ -- we've already written  
84 ÏÏ§ÏÏ©end ">=";  
85 ÏÏ§  
86 ÏÏ§----------------------------------------------------------------------------

-- function >= decides if all the members of set right belong to set left

-- returns true or false, depending  
87 ÏÏ§ÏÞßàfunction ">" (Left : in Set\_Type; Right : in Set\_Type) return Boolean is  
88 ÏÏ§ÏÏ§begin  
89 ÏÏ§Â¹Ä¹¹Ïreturn Right < Left; -- Reverse the order of the   
90 ÏÏ§ÏÏ§ -- parameters and use the operation   
91 ÏÏ§ÏÏ§ -- we've already written  
92 ÏÏ§ÏÏ©end ">";  
93 ÏÏ§  
94 ÏÏ©end Discrete\_Set;  
95