1 ÏÕÖ×package body Discrete\_Set is  
 2 ÏÏ§  
 3 ÏÏ§----------------------------------------------------------------------------  
 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); --Set(Element) is a Boolean. If Element

-- exists in Set, then True is returned. If

-- it doesn’t exist, then it returns False.  
 8 ÏÏ§ÏÏ©end Is\_Member;  
 9 ÏÏ§  
10 ÏÏ§----------------------------------------------------------------------------  
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; --Returns a set containing every element

-- that is found in either

-- the left or the right set, or in both.  
14 ÏÏ§ÏÏ©end "+";  
15 ÏÏ§  
16 ÏÏ§----------------------------------------------------------------------------  
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; --The set to be returned  
20 ÏÏ§ÏÏ§begin  
21 ÏÏ§ÏÏ¨¹¹ÏResult := Left; --The set that will be returned now contains the

-- elements of the Left set  
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§----------------------------------------------------------------------------  
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; --The set to be returned  
30 ÏÏ§ÏÏ§begin  
31 ÏÏ§ÏÏ¨¹¹ÏResult := Right; --The set that will be returned now contains the

-- elements of the Right set  
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§----------------------------------------------------------------------------  
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; --Returns a set of only the elements which

-- are found in both the Left and the Right

-- sets  
40 ÏÏ§ÏÏ©end "\*";  
41 ÏÏ§  
42 ÏÏ§----------------------------------------------------------------------------  
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; --Returns a set of elements which belong

-- only to the Left set; any elements

-- found in Left that are also found in

-- Right are not part of the set returned  
46 ÏÏ§ÏÏ©end "-";  
47 ÏÏ§  
48 ÏÏ§----------------------------------------------------------------------------

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; --The set that will be returned  
52 ÏÏ§ÏÏ§begin  
53 ÏÏ§ÏÏ¨¹¹ÏResult := Left; --The set that will be returned now contains

-- the elements from the Left set  
54 ÏÏ§ÏÏ¨¹¹ÏResult(Right) := False; -- Remove the element from the set  
55 ÏÏ§Â¹Ä¹¹Ïreturn Result;  
56 ÏÏ§ÏÏ©end "-";  
57 ÏÏ§  
58 ÏÏ§----------------------------------------------------------------------------

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; --Is\_A\_Subset will only be

-- true if every element

-- from the Left set is

-- also an element found in

-- the Right set  
63 ÏÏ§Â¹Ä¹¹Ïreturn Is\_A\_Subset;  
64 ÏÏ§ÏÏ©end "<=";  
65 ÏÏ§  
66 ÏÏ§----------------------------------------------------------------------------  
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§----------------------------------------------------------------------------  
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§----------------------------------------------------------------------------  
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