Table 4.2: *How the different classes in the European hierarchy implement the greet() operation*

Class	*Method specification for greet()*
European	undefined
Briton	Good morning
Frenchman	Bonjour
German	Guten Tag
Italian	Buongiorno

implemented by a number of different methods is called *polymorphic.*

Polymorphic operations are used in the context of an inheritance hierarchy where the same operation may be implemented differently in each subclass. A single message will produce a different response depending on the class of the object to which it is sent. For example, in the inheritance hierarchy shown in Figure 4.19 the operation greet() appears in all of the classes in the hierarchy. The method for greet() is undefined in the class European.[6] Each of the other classes in the hierarchy has a different method for greet() (see Table 4.2).

To illustrate polymorphism in action, let us create an object called pierre of the Frenchman class with French as his language. If we send him the message pierre.greet() he will respond by saying 'Bonjour', see Table 4.3 which also shows the response of hans, a :German. We have set you an exercise to work out the responses of george, a :Briton, and antonio, an :Italian (see Exercise 4.9)

To understand a bit more how polymorphism works, we create an array of four European objects, named nationality[4]. We can populate this with objects of all of the instantiable classes in the European hierarchy as in Figure 4.21.

We can iterate through the array sending the greet() message to each of the objects in turn (see Figure 4.22); each will respond according to the method implementation of their class, and the

6. *The reason for putting any undefined operation into a superclass (as we have done with greet()) is to try to build some future-proofing into the model; we are guessing that if new classes are introduced to the hierarchy, they will need this operation. If a new class is introduced it will inherit this operation and define it to suit its purpose. This should simplify the process of modifying the system.*