

```

    theHand += 11;
    handContainsAce11 = true;
}
// In all other cases, the card's face value is added
else
    theHand += theCard;

// If the new hand value is over 21, try to shrink it
if( theHand > 21 && handContainsAce11 )
{
    theHand -= 10;
    handContainsAce11 = false;
}

return theHand;
}

```

Note that the function accepts the hand's initial value as an input parameter and returns the hand's new value as the function's output value. It would have been perfectly acceptable to make the hand's value a reference parameter instead. (In fact, some programmers would have made the hand's value a reference *and* returned it from the function.)

SUMMARY

With multiple functions, reference parameters, prototypes, and header files, it is possible to create projects of moderate size that are easy to organize and understand.

However, Blackjack is a very special case among card games, because with the exception of the "big aces" that we discussed earlier, the identities of individual cards are unimportant once they have been added to the hand; only the hand's aggregate value matters. In a more traditional game like Rummy, Poker, or even Crazy Eights, it is crucial to know the identity of each card in a player's hand at all times. This cannot be done easily with the simple variables you have been using since the beginning of this book—in Part IV, you will add data structures to your arsenal, thus introducing a great deal of additional flexibility into our programming.