

### Dealing with Ace's

- In BlackJack, Ace can be 1 or 11. Crucial observation: there can only be at most one 11: two 11 adds up to more than 21.
- So the easy way to handle Ace is to treat it as 1, and remember that there is an Ace.
- So we keep track of the "minimum point" of a hand and also whether it contains an Ace. That's enough for us to know what is the number of points of a hand.
- E.g. 1: A, A, 2, 3, 6 has a "minimum point" of  $1+1+2+3+6=13$ . Adding 10 results in 23. So although it has an Ace, we cannot add 10 to it.
- E.g. 2: A, 2, 2, 3 has a "minimum point" of  $1+2+2+3=8$ . Adding 10 results in 18. So we add 10 to get the "real" point of 18.

0911B L22-1

### One more class

- Now suddenly we want one more class, to hold the status of the player: whether we already have an ace, and what is the minimum point for our hand.
- Let's also make it provide two services: to add one card into it, and to get the value:
  - public void add(PlayingCard card)
  - public int point()
- Note that both are instance methods: given such an object, we want to add a card to it, or to get the value of it.
- It's a really trivial class, let's write it right away.

0911B L22-2

### Black Jack

The structure of our program:

- We have a class variable of type RandomDeck containing the deck, every method go to this deck to get a card.
- We have two class variables of type PlayerStat containing the status of the dealer and the player.
- Both the dealer and the player need to draw card, print out the card and also to add it to the PlayerStat object. We make a method giveCard() for it, giving it an argument whether to give card to the dealer or the player.
- We have a method for the player's turn. Everything else is put into the main method. The main method thus decides which side win and print out such messages.

0911B L22-3

### Main engine

The next task is the Black Jack engine. Top down design, of course.

```
Get a card for the dealer, show the card
Let the player to get cards
if (the player has more than 21 points) {
    Print dealer winning.
} else {
    while (dealer's point is less than 16)
        give dealer a card
    Show values of both sides
    if (the dealer has more than 21 points or less than the player) {
        Print player winning.
    } else {
        Print dealer winning.
    }
}
```

0911B L22-4

### Player's turn

On player's turn, we basically continue to ask him whether a card is desired.

```
give player a card
give player a card
ask player whether he/she want more
while (he/she answer yes) {
    give player a card
    if player get at least 21 points
        return
    ask player whether he/she want more
}
```

0911B L22-5