B0544222 李丞翊 資管二乙

```
8       private int bet = 10;
9       private int[] lastbet = { 10, 10 };
10      @Override
11      public int make_bet() {
12          if (lastbet[0] == getBet()) {
13              if (lastbet[1] >= 50)
14                  bet = lastbet[1] - 10;
15              else
16                  bet = lastbet[1] * 3;
17          } else {
18              if (lastbet[0] >= 50)
19                  bet = lastbet[0] - 10;
20              else
21                  bet = lastbet[0] * 3;
22          }
23          setBet(bet);
24
25          if (getBet() > get_current_chips()) {
26              return get_current_chips();
27          } else {
28              return getBet();
29          }
30      }
```

(圖一)以上為程式碼

在“下注”的策略：

以跟注的方式=>前一個玩家投入賭金後，第二位玩家也下了與第一位玩家相同的賭注，讀取另一位玩家上一回的下注，再去判斷自己這回合的下注方式

註解第一行：起始 開始的賭注為 10
預設剛開始第一回合的下注皆為 10
當要決定下著的時候：

如果 這次要下的注和上一次下的注一樣的話，判定你讀到自己的，
就 讀 另一個人的注，再去判斷 如果他的注大於等於 50 的話，我下的注將是他的注-10（保守一點），而小於 50 的話將他的賭注乘以三倍，而設定 50 當成判斷的依據，是因為普通同學都會因為有兩百局和籌碼只有 1000 前的限制下，所下注的籌碼並不會太高，因此大概設定 50 左右，在 10 和 3 的參數，是經過大量測試後，平均籌碼最大的一組，因此採用。

若和上一次的注不一樣的話，直接判斷自己上次是否大於 50，再做運算。
第 25 行開始的部分是判斷下注是否有大於 目前所擁有的籌碼。
若有的話，就決定梭哈;沒有就回傳下注的值。

```
32      @Override
33      public boolean hit_me(Table tbl) {
34          lastbet = tbl.get_palyers_bet();//取得玩家的賭注
35
36          int dealer = tbl.get_face_up_card_of_dealer().getRank();
37          boolean hit = false;
38
39          if (getTotalValue() <= 11)
40              hit = true;
41           else {
42              if (getTotalValue() <= 16 && (dealer >= 6 || dealer == 1))
43                  hit = true;
44          }
45          return hit;
46      }
```

(圖二)以上為程式碼

在“要牌”的策略：

如果本身的牌小於 11 的話，就決定要牌（因為再加一張牌也不會爆掉），而另一種情況是本身的總點數小於等於 16 的時候，表示自己的牌是介於 12~16 之間，那這時候我可以看到莊家的第一張牌，去判斷莊家第一張大於等於 6 的時候（表示它可能是 6,7,8,9,10 的牌）或是第一張牌是一的時候(因為 1 可以當 11)也算是大牌，這個時候我認為他贏的機會變大了，那我目前點數比他小，我如果不要牌的話會輸，乾脆“要一張牌”去拼拼看會不會比他大，如果爆掉的話就輸掉這一局，但是贏的機會比較大（手排是介於 12~16 之間，下一張的可能是 1~10 之間），有 3/10～5/10 的機率會爆掉，並沒有超過一半，反而表示有 5/10~7/10 的機率會贏。