

```

public static class BanditUCBPlayer extends Player {

    int num_plays;
    HashMap bet = new HashMap();
    HashMap fold = new HashMap();

    public void reset() {
        num_plays = 1;
        bet.clear();
        fold.clear();
    }

    public boolean getBet(CardExt card1, CardExt card2) {
        String hash_value = card1.toString() + card2.toString();
        BanditResult r_bet = null;
        BanditResult r_fold = null;
        if ((r_bet = (BanditResult)bet.get(hash_value)) == null) {
            r_bet = new BanditResult();
            bet.put(hash_value, r_bet);
            r_fold = new BanditResult();
            fold.put(hash_value, r_fold);
        } else {
            r_fold = (BanditResult)fold.get(hash_value);
        }

        double bet_value = (r_bet.total / r_bet.tries) +
            Math.sqrt((2.0 * Math.log(num_plays))/(r_bet.tries));
        double fold_value = (r_fold.total / r_fold.tries) +
            Math.sqrt((2.0 * Math.log(num_plays))/(r_fold.tries));

        //System.out.println("Bet: " + bet_value);
        //System.out.println("Fold:" + fold_value);

        return (bet_value >= fold_value);
    }

    public void processResult(CardExt[] hand1, CardExt[] hand2,
        int win_loss, boolean did_bet) {
        num_plays++;
        String hash_value =
            hand1[0].toString() + hand1[1].toString();
        BanditResult r = (BanditResult)
            ((HashMap)(did_bet ? bet : fold)).get(hash_value);
        r.tries++;
        r.total += win_loss;
    }
}

```