

# DynamicWebProgramming HW3

## 112550020 蔡懷恩

### 遊戲規則及流程

1. 每回合開始前，玩家需先下注，可以選擇下注金額，下注完，按鈕會灰掉不能按。



```
// ---- 下注 ----
function setBet() {
    if (isBetLocked) {
        showMessage("本局已下注，請先完成遊戲再下注。");
        return;
    }
    const betInput = document.getElementById("bet-input");
    const betValue = parseInt(betInput.value);

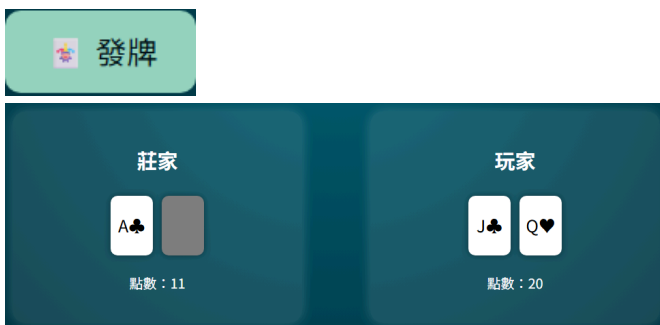
    if (isNaN(betValue) || betValue <= 0) {
        showMessage("請輸入正確的下注金額。");
        return;
    }

    if (betValue > playerMoney) {
        showMessage("金額不足。");
        return;
    }

    bet = betValue;
    playerMoney -= betValue; // 扣除下注金額
    document.getElementById("bet-btn").disabled = true;
    document.getElementById("deal-btn").disabled = false;
    isBetLocked = true;
    updateMoneyDisplay();
    showMessage(`你下注了 ${bet}! 按「發牌」開始。`);
}
```

2. 按下發牌開始遊戲，系統會自動發給：

- 玩家：兩張明牌
- 莊家：一張明牌、一張暗牌



```
// ---- 遊戲開始 ----
function deal() {
    if (!isBetLocked) {
        showMessage("本局還未下注，請先下注再遊玩。");
        return;
    }
    document.getElementById("hit-btn").disabled = false;
    document.getElementById("stand-btn").disabled = false;
    createDeck();
    shuffleDeck();

    if (isCheated) {
        cheatMoveAand10ToTop();
    }

    dealerHand = [drawCard(), drawCard()];
    playerHand = [drawCard(), drawCard()];

    updatePoints(false);
    renderHands(false); // false 表示莊家有一張是蓋住的
}
```

3. 玩家可以選擇：

- 要牌(Hit):再抽一張, 直到點數超過 21(爆牌)
- 停牌(Stand):結束回合



```
// ---- 要牌 ----
async function hit() {
  playerHand.push(drawCard());
  updatePoints(false);
  renderHands(false);
  // 延遲 0.5 秒，避免還沒更新就顯示訊息
  await new Promise(r => setTimeout(r, 300));

  if (playerPoints > 21) {
    renderHands(true);
    updatePoints(true);
    showMessage("你爆了！莊家勝利。");
    saveGameResult();
    isBetLocked = false;
    document.getElementById("bet-btn").disabled = false;
    document.getElementById("deal-btn").disabled = true;
    document.getElementById("hit-btn").disabled = true;
    document.getElementById("stand-btn").disabled = true;
  }
}
```

4. 莊家回合：

- 停牌後，翻開暗牌，若總點數 < 17 則自動補牌；直到  $\geq 17$  則停止。



```
// ---- 停牌 ----
async function stand() {
  // 莊家補牌直到  $\geq 17$ 
  renderHands(true);
  updatePoints(true);
  while (dealerPoints < 17) {
    dealerHand.push(drawCard());
    renderHands(true);
    updatePoints(true);
    await new Promise(r => setTimeout(r, 300));
  }

  checkWinner();
}
```

## 更新點數

```
function updatePoints(showDealerAll) {
  playerPoints = calculatePoints(playerHand);
  document.getElementById("player-points").innerText = "點數：" + playerPoints;

  if (showDealerAll) {
    // 翻牌後才算全部
    dealerPoints = calculatePoints(dealerHand);
  } else {
    // 只算第一張明牌
    dealerPoints = calculatePoints([dealerHand[0]]);
  }
  document.getElementById("dealer-points").innerText = "點數：" + dealerPoints;
}
```

## 顯示牌

```
function renderHands(showDealerAll) {
  const playerDiv = document.getElementById("player-cards");
  const dealerDiv = document.getElementById("dealer-cards");
  playerDiv.innerHTML = "";
  dealerDiv.innerHTML = "";

  // 顯示玩家手牌
  playerHand.forEach(card => {
    playerDiv.innerHTML += `<div class='card'>${card.value}${card.suit}</div>`;
  });

  // 顯示莊家手牌
  dealerHand.forEach((card, i) => {
    if (!showDealerAll && i === 1) {
      dealerDiv.innerHTML += `<div class='card hidden'>🀄</div>`;
    } else {
      dealerDiv.innerHTML += `<div class='card'>${card.value}${card.suit}</div>`;
    }
  });
}
```

## 5. 勝負判定:

- 莊家爆牌 → 玩家勝 → 贏得兩倍賭注
- 玩家爆牌 → 莊家勝 → 賭注扣除
- 雙方未爆牌 → 點數高者勝 → 贏得兩倍賭注 / 賭注扣除
- 點數相同 → 平手 → 退回賭注

```
function checkWinner() {
  let result = "";
  if (dealerPoints > 21) {
    result = '莊家爆了！你贏了！';
    playerMoney += bet * 2;
  }
  else if (playerPoints > 21) {
    result = '你爆了！莊家勝利。';
  }
  else if (playerPoints > dealerPoints) {
    result = '你贏了！';
    playerMoney += bet * 2;
  }
  else if (playerPoints < dealerPoints) {
    result = '莊家贏了！';
  }
  else {
    result = '平手。';
    playerMoney += bet; // 退錢
  }

  updateMoneyDisplay();
  showMessage(result);
  saveGameResult();
}
```



## 6. 紀錄結果:

紀錄每輪結束時，  
在Local Storage 存遊玩時間、分數、餘額，  
在Session Storage 存牌面

```
function saveGameResult() {
  const now = new Date();
  const timeStr = now.toLocaleString();

  // === LocalStorage ===
  const localData = JSON.parse(localStorage.getItem("blackjack_local") || '{"rounds":[]}'); // 用 || 避免沒資料
  const Round = localData.rounds.length + 1;

  localData.rounds.push({
    round: Round,
    playerPoints: playerPoints,
    dealerPoints: dealerPoints,
    playerMoney: playerMoney, // 你可以自行定義初始值 1000 + 輸贏加減
    playTime: timeStr
  });

  localStorage.setItem("blackjack_local", JSON.stringify(localData));

  // === SessionStorage ===
  const sessionData = JSON.parse(sessionStorage.getItem("blackjack_session") || '{"rounds":[]}');
  sessionData.rounds.push({
    round: Round,
    playerCards: playerHand.map(c => `${c.value}${c.suit}`),
    dealerCards: dealerHand.map(c => `${c.value}${c.suit}`)
  });
  sessionStorage.setItem("blackjack_session", JSON.stringify(sessionData));
}
```

## 顯示結果

遊戲紀錄						
回合	遊玩時間	玩家點數	莊家點數	剩下金額	莊家手牌	玩家手牌
1	2025/11/13 上午 4:08:15	19	25	1100	10♥ 6♥ 9♦	5♥ 2♥ 6♠ 2♣ 4♠
2	2025/11/13 上午 4:16:02	23	20	1000	10♥ J♦	5♥ 3♣ 10♠ 2♣ 3♥
3	2025/11/13 上午 4:16:35	26	8	900	4♥ 4♣	2♥ 7♦ 7♥ 10♥
4	2025/11/13 上午 4:16:43	22	17	800	7♠ 10♥	10♣ 2♣ 10♦

```
function showRecords() {
  const tableBody = document.querySelector("#record-table tbody");
  tableBody.innerHTML = ""; // 清空舊資料

  const localData = JSON.parse(localStorage.getItem("blackjack_local") || '{"rounds":[]}');
  const sessionData = JSON.parse(sessionStorage.getItem("blackjack_session") || '{"rounds":[]}');

  if (localData.rounds.length === 0 && sessionData.rounds.length === 0) {
    showMessage("目前沒有任何紀錄。");
    return;
  }

  localData.rounds.forEach(lr => {
    const sr = sessionData.rounds.find(s => s.round === lr.round);

    const row = document.createElement("tr");
    row.innerHTML = `
      <td>${lr.round}</td>
      <td>${lr.playTime}</td>
      <td>${lr.playerPoints}</td>
      <td>${lr.dealerPoints}</td>
      <td>${lr.playerMoney ?? "-"}</td>
      <td>${sr ? sr.dealerCards.join(" ") : "-"}</td>
      <td>${sr ? sr.playerCards.join(" ") : "-"}</td>
    `;
    tableBody.appendChild(row);
  });

  showMessage("已載入紀錄！");
}
```

 重設 重設按鈕

清除local storage 和 session storage的東西  
並把所有數值都初始化，最後跳出視窗。



## 作弊模式

關閉



開啟



```
function updateCheat(){
  const btn = document.getElementById("cheat-btn");
  if (isCheated) {
    isCheated = false;
    btn.classList.add("off");
    btn.classList.remove("on");
  } else {
    isCheated = true;
    btn.classList.add("on");
    btn.classList.remove("off");
  }
}
```

找出一張 A 和一張 10, 並把他們放到牌組最上層, 讓莊家先抽,  
即可讓莊家每次都達成21點, 進而每次都獲勝或平手

```
function cheatMoveAand10ToTop() {
  // 找出第一張 A 和第一張 10
  const aceIndex = deck.findIndex(c => c.value === 'A');
  const tenIndex = deck.findIndex(c => c.value === '10');

  if (aceIndex === -1 || tenIndex === -1) {
    showMessage("找不到 A 或 10, 無法作弊。");
    return;
  }

  // 取出 A 和 10
  const aceCard = deck.splice(aceIndex, 1)[0];
  const tenCard = deck.splice(tenIndex > aceIndex ? tenIndex - 1 : tenIndex, 1)[0]; // 10 在 A 後面的話 index 會換

  // 把它們放到最上面 (讓莊家拿到)
  if (Math.random() < 0.5) {
    // A 在前
    deck.push(aceCard);
    deck.push(tenCard);
  } else {
    // 10 在前
    deck.push(tenCard);
    deck.push(aceCard);
  }
}
```