1. 最後需交出一個 **function** 叫 **craps_game** ， 該 **function** 有兩個輸入：**num_simulations (the number of simulations to run)** 和 **bet_size (the size of the bet for each game)**。**function** 裡為模擬此賭博遊戲的輸贏。

```
# 依照題意列出函數結構
# craps_game <- function(num_simulations, bet_size) { }
```

2. 此賭博遊戲為，用 **sample()**去模擬擲兩個骰子。加總擲出的兩個骰子數字，為點數和。

```
# 建立一個 function 叫 point，會回傳加總值出的兩個骰子點數和
# 透過 sample(1:6,2)隨機取得兩個 1~6 之間的整數，並透過 sum()將 sample 取得的數字做加總
point <- function() {
  return (sum(sample(1:6,2)))
}

# 執行多次 point()的結果，應為 2~12 常態分布
```

```
> point()
[1] 4
```

3. 如果第一次點數和為 **7** 或 **11**，玩家勝，贏得 **bet size** 。 若點數和為 **2**、**3** 或 **12**，玩家輸，輸了 **-1 x bet size**。如果點數和為其他點數，記錄第一次的點數和，然後繼續擲骰，直至點數和等於第一次擲出的點數和，玩家勝，贏得 **bet size**。若在這之前擲出了點數和為 **7**，玩家輸，輸了 **-1 x bet size**。

```
# 建立一個 function 叫 if_win，依照上述規則判斷是否贏得 bet_size
if_win <- function() {
    # 先透過前面寫的 piont()取得第一次點數和
    first_point <- point()
    # 如果點數和為 7 或 11，回傳 TRUE 代表贏得遊戲
    if (first_point == 7 || first_point == 11) {
        return (TRUE)
    # 如果點數和為 2、3 或 12，回傳 FALSE 代表輸了
    } else if (first_point == 2 || first_point == 3 || first_point == 12) {
        return (FALSE)
    } else {
        # 如果為其他點數，先記錄第一次點數和
        target <- first_point
        retry_point <- 0
        # 繼續擲骰直到點數和等於第一次或 7
        while (retry_point != 7 && retry_point != target) {
            retry_point <- point()
        }
```

```
        # 如果等於第一次和，回傳 TRUE 代表贏得遊戲
        if (retry_point == target) {
            return (TRUE)
        # 如果等於 7 和，回傳 FALSE 代表輸了
        } else if (retry_point == 7) {
            return (FALSE)
        }
    }
}
```

```
> if_win()
[1] FALSE
```

4. **計算遊戲的期望值**

   expected value = (probability of winning x bet size) + (probability of losing x bet size).

```
# 建立一個 function 叫 expected_value，需傳入贏的機率和賭注金額兩個參數
# 透過 probability_of_winning*bet_size 取得贏的期望值
# 只有輸贏兩種可能，所以透過 1 - probability_of_winning 取得輸的機率
# 輸的金額是負的 bet_size，所以贏的期望值減輸的期望值就是整個遊戲的期望值
# 可以再簡化成 return ((2*probability_of_winning-1)*bet_size)
# 但考量到便於理解，以及可能需要調整輸贏結果，不再做進一步簡化
expected_value <- function(probability_of_winning, bet_size) {
    return (probability_of_winning*bet_size - (1-probability_of_winning)*bet_size)
}

# 放機率 0.6 跟賭注金額 10，用來測試函數
```

```
> expected_value(0.6, 10)
[1] 2
```

5. **craps_game function 輸出（Return） 遊戲的期望值。**

   提示：先寫玩一次的 function，贏為 TRUE，輸為 FALSE，輸出該結果。再放進 craps_game 重複多次後，計算輸贏次數，獲得輸贏機率。

```
# 建立一個 function 叫 craps_game，如第一題所述需傳入測試次數和賭注金額兩個參數
craps_game <- function(num_simulations, bet_size) {
    # 建立一個變數叫 win_count
    win_count <- 0
```

```
    # 依照傳入參數重複執行 if_win()，如果回傳值為正，win_count+1
    for (i in c(1:num_simulations)) {
        if (if_win()) {
            win_count <- win_count + 1
        }
    }
    # 贏的機會 = 前面算出來的 win_count 除以總執行次數
    probability_of_winning <- win_count/num_simulations
    # 透過前面寫的 expected_value 來取得期望值
    return(expected_value(probability_of_winning, bet_size))
}

# 放執行 1000 次跟賭注金額 10，用來測試函數
> craps_game(1000, 10)
[1] 0.14
```