

Final Project Report

Project 1

合約地址: 0xd6df1978d26f378c0813b8f67ce7547083cf1da1

Logic

接受不同玩家的不同投注(大於entranceFee就好)，並且玩家投注為累積的，可以重複投注，勝率與該玩家總投注金額正比

Example:

entranceFee = 1

A先投3

B投5

C投2

A投2

結算時:

總金額 = $3 + 5 + 2 + 2 = 12$

A總金額 $3 + 2 = 5$

B總金額5

C總金額2

index = 亂數 % 12 = 贏家的money

index 落在0~4: A贏

index 落在5~9: B贏

index 落在10~11: C贏

Structure and Member

An enum for status

```
1 enum lotteryStatus{
2     NO, // not playing
3     PLAYING, // playing
4     COUNTING // counting result
5 }
```

members

```
1 mapping(uint256=>address) private IDtoPlayer; // to emulate the
   traverse of players mapping
2 mapping(address=>uint256) private players; // 玩家到投注金額的mapping
3
4 uint256 public totalPlayer; // 全部參與的玩家
5 address public recentWinner; // 最新的贏家地址
6 uint256 public entranceFee; // 每次投注的threshold
7
8 address private _owner; // 合約持有者
9 uint256 private _totalMoney; // 總投注金額
10 lotteryStatus private _STATUS; // 現在遊戲的狀態
```

Modifier

To check if the function is execute by the owner

```
1 modifier isOwner(){
2     require(_owner == msg.sender);
3     _;
4 }
```

constructor

The constructor setting the onwer and status of the game to no

```
1 constructor() public {
2     _owner = msg.sender;
3     _STATUS = lotteryStatus.NO;
4 }
```

functions

```
1 function enter() public payable {
2     // 檢查狀態是不是lotteryStatus.PLAYING跟輸入金額是否大於entranceFee
3     require(_STATUS == lotteryStatus.PLAYING);
4     require(msg.value >= entranceFee);
5
6     // 把總金額+輸入金額
7     _totalMoney += msg.value;
8
9     // 如果玩家不存在，則新增玩
10    if(players[msg.sender] == 0)
11        IDtoPlayer[totalPlayer++] = msg.sender;
12
13    // 把金額送到該玩家的mapping
14    players[msg.sender] += msg.value;
15 }
```

```
1 // 檢查是否是owner執行
2 function startLottery(uint256 fee) public isOwner {
3     // 檢查狀態是否是沒在玩的狀態
4     require(_STATUS == lotteryStatus.NO);
5
6     // 設定入場費
7     entranceFee = fee;
8
9     // 把玩家數跟總金額歸零
10    totalPlayer = 0;
11    _totalMoney = 0;
12 }
```

```
13      // 改變狀態為正在遊玩
14      _STATUS = lotteryStatus.PLAYING;
15  }
```

```
1  // 檢查是否是owner執行
2  function endLottery() public isOwner{
3      // 檢查狀態是否是正在玩的狀態
4      require(_STATUS == lotteryStatus.PLAYING);
5
6      // 狀態設定為正在結算
7      _STATUS = lotteryStatus.COUNTING;
8
9      // 用亂數生成找出贏家的是介於哪個money區段的
10     uint256 indexOfWinner =
        uint256(keccak256(abi.encodePacked(msg.sender, block.difficulty,
        block.timestamp))) % _totalMoney;
11
12     // 目前累積金額跟贏家的index
13     uint256 cur = 0;
14     uint256 i;
15
16
17     // 遍歷所有玩家並用IDtoPlayer找出現在的index是哪個address，再用找到的
        address去找到目前金額
18     // 如果大於目前金額代表這個index的money落在這個玩家的區間段，因此由這個玩家獲
        勝
19     for(i = 0; i < totalPlayer; ++i){
20         cur += players[IDtoPlayer[i]];
21         if(cur >= indexOfWinner)
22             break;
23     }
24
25     // 設定贏家
26     recentWinner = IDtoPlayer[i];
27
28     // 轉錢過去並確定是否成功
```

```
29     (bool finish, ) = payable(recentWinner).call{value: _totalMoney}
    ("");
30     require(finish == true);
31
32
33     // 把所有參賽者金額歸零(清空mapping)
34     for(i = 0; i < totalPlayer; ++i)
35         players[IDtoPlayer[i]] = 0;
36
37     // 狀態設定為沒有在玩
38     _STATUS = lotteryStatus.NO;
39
40 }
```

執行結果

startLottery:



[vm] from: 0x5B3...eddC4
to: Lottery.startLottery(uint256) 0xd91...39138 value: 0 wei
data: 0xc6b...003e8 logs: 0 hash: 0xe38...aa197

Debug



status	true Transaction mined and execution succeed
transaction hash	0xe3821c8c1b9c433f1816964f135bb21045ebe535d42dce18a4bfd726f69aa197 🔗
from	0x5B380a6a701c568545dCfcB03FcB875f56beddC4 🔗
to	Lottery.startLottery(uint256) 0xd9145CCE52D386f254917e481eB44e9943F39138 🔗
gas	83715 gas 🔗
transaction cost	72795 gas 🔗
execution cost	72795 gas 🔗
input	0xc6b...003e8 🔗
decoded input	{ "uint256 fee": "1000" } 🔗
decoded output	{ } 🔗
logs	[] 🔗 🔗
val	0 wei 🔗

enter:

1. 失敗(小於entreeFee)



[vm] from: 0x5B3...eddC4 to: Lottery.enter() 0xd91...39138
value: 999 wei data: 0xe97...dcb62 logs: 0 hash: 0x8ed...7ca0a

Debug



status	false Transaction mined but execution failed
transaction hash	0x8ed703c78858c89f28b596d0d14f20386d94a4dd5abb9e29f8a448d90e57ca0a 🔗
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 🔗
to	Lottery.enter() 0xd9145CCE52D386f254917e481eB44e9943F39138 🔗
gas	3000000 gas 🔗
transaction cost	25601 gas 🔗
execution cost	25601 gas 🔗
input	0xe97...dcb62 🔗
decoded input	{ } 🔗
decoded output	{ } 🔗
logs	[] 🔗 🔗
val	999 wei 🔗

transact to Lottery.enter errored: VM error: revert.

revert

The transaction has been reverted to the initial state.

Note: The called function should be payable if you send value and the value you send should be
Debug the transaction to get more information.

2. 成功



[vm] from: 0x5B3...eddC4 to: Lottery.enter() 0xd91...39138
value: 1001 wei data: 0xe97...dcb62 logs: 0 hash: 0x8f2...86c7b

Debug



status	true Transaction mined and execution succeed
transaction hash	0x8f221a779b993c34b91355c9c596affe612b75944a2a4ba95a76437462386c7b 🔗
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 🔗
to	Lottery.enter() 0xd9145CCE52D386f254917e481eB44e9943F39138 🔗
gas	132306 gas 🔗
transaction cost	115048 gas 🔗
execution cost	115048 gas 🔗
input	0xe97...dcb62 🔗
decoded input	{ } 🔗
decoded output	{ } 🔗
logs	[] 🔗 🔗
val	1001 wei 🔗

3. 成功(使用不同人投注):



[vm] from: 0xAb8...35cb2 to: Lottery.enter() 0xd91...39138
value: 1005 wei data: 0xe97...dcb62 logs: 0 hash: 0xbad...d6883

Debug



status	true Transaction mined and execution succeed
transaction hash	0xbad1c9c7a5dce3cdd73d67626984c9ff631f8537a64e7c559c42a4c6941d6883 🔗
from	0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2 🔗
to	Lottery.enter() 0xd9145CCE52D386f254917e481eB44e9943F39138 🔗
gas	92976 gas 🔗
transaction cost	80848 gas 🔗
execution cost	80848 gas 🔗
input	0xe97...dcb62 🔗
decoded input	{ } 🔗
decoded output	{ } 🔗
logs	[] 🔗 🔗
val	1005 wei 🔗

查詢入場費用:

CALL

[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to: Lottery.entranceFee() data: 0x649...677e1

Debug

from

0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to

Lottery.entranceFee() 0xd9145CCE52D386f254917e481eB44e9943F39138

execution cost

23604 gas (Cost only applies when called by a contract)

input

0x649...677e1

decoded input

{}

decoded output

{
 "0": "uint256: 1000"
}

logs

[]

查詢入場人數:

CALL

[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to: Lottery.totalPlayer() data: 0xc49...4a080

Debug

from

0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to

Lottery.totalPlayer() 0xd9145CCE52D386f254917e481eB44e9943F39138

execution cost

23537 gas (Cost only applies when called by a contract)

input

0xc49...4a080

decoded input

{}

decoded output


{
 "0": "uint256: 2"
}

logs









[]

endLottery:

- 1. 失敗(非owner執行)

 [vm] from: 0xAb8...35cb2 to: Lottery.endLottery() 0xd91...39138
value: 0 wei data: 0x159...3a8c7 logs: 0 hash: 0xfec...0b78c

Debug ^

status	false Transaction mined but execution failed
transaction hash	0xfec1dbecc1ab21d3aff9d5caf1b6825963b600f30e781defdbe8fa051910b78c 
from	0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2 
to	Lottery.endLottery() 0xd9145CCE52D386f254917e481eB44e9943F39138 
gas	3000000 gas 
transaction cost	23397 gas 
execution cost	23397 gas 
input	0x159...3a8c7 
decoded input	{ } 
decoded output	{ } 
logs	[]  
val	0 wei 

transact to Lottery.endLottery errored: VM error: revert.

revert

The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the value you send should be
Debug the transaction to get more information.

2. 成功



[vm] from: 0x5B3...eddC4 to: Lottery.endLottery() 0xd91...39138
value: 0 wei data: 0x159...3a8c7 logs: 0 hash: 0xc53...13eae

Debug



status	true Transaction mined and execution succeed
transaction hash	0xc5327f3e131f4b96ebafa2725a94d2e82e2b9b217a812d70b3a1f9d4c4913eae 🔗
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 🔗
to	Lottery.endLottery() 0xd9145CCE52D386f254917e481e844e9943F39138 🔗
gas	276313 gas 🔗
transaction cost	225872 gas 🔗
execution cost	225872 gas 🔗
input	0x159...3a8c7 🔗
decoded input	{ } 🔗
decoded output	{ } 🔗
logs	[] 🔗 🔗
val	0 wei 🔗

Project2

只列出與Project1不同的地方

Logic

1. 每次結算都記錄10%金額，在增加一個Function讓owner執行輸入提領的地址並進行提領
2. 提供publicMember查看
3. 提供publicMember查看
4. 提供publicMember查看
5. 新增一個struct存贏家address, 金額跟人數，然後用array包起來提供function輸入第幾輪來查詢
6. 新增一個mapping(慈善機構address=>每次提領的紀錄的陣列)，新增一個function,給定地址讓玩家輸入慈善機構address查詢前面提到的mapping

Structure and Member

```
1  // 紀錄每個round的狀態
2  struct roundStatus{
3      address winnerAddress;
4      uint256 winnerAmount;
5      uint256 totalPlayer;
6  }
7
8  RoundStatus[] private roundStatus; // 每個round的陣列
9
10 mapping(address=>uint256[]) private charityRecord; // 提領紀錄
11
12 uint256 public totalHistoryPlaytime; // 歷史總開獎次數
13 uint256 public totalHistoryMoney; // 歷史總金額
14 uint256 public totalHistoryPlayer; // 歷史總玩家
15
16 uint256 private _totalCharity; // 全部的慈善金額
```

functions



```
1  constructor() public{
2      // 初始化數值
3      totalHistoryPlaytime = 0;
4      totalHistoryMoney = 0;
5      totalHistoryPlayer = 0;
6      _totalCharity = 0;
7  }
8
9  // 給定某慈善機構地址進行提領
10 function withDraw(address addr) public isOwner {
11     // 提領並檢查
12     (bool finish, ) = payable(addr).call{value: _totalCharity}("");
13     require(finish == true);
14 }
```













```
15      // 把提領紀錄到charityRecord
16      charityRecord[addr].push(_totalCharity);
17
18      // 歸零
19      _totalCharity = 0;
20  }
21
22
23  function endLottery() public isOwner{
24
25
26      // 處理能查詢的變數
27      ++totalHistoryPlaytime;
28      totalHistoryMoney += _totalMoney;
29      totalHistoryPlayer += totalPlayer;
30
31      // 玩家拿9/10的總金額
32      uint256 totalPay = _totalMoney * 9 / 10;
33      (bool finish, ) = payable(recentWinner).call{value: totalPay}("");
34      require(finish == true);
35
36      // 把這輪狀態記下來
37      roundStatus.push(RoundStatus(recentWinner, totalPay, totalPlayer));
38
39      // 剩下的累積到慈善機構總額
40      _totalCharity += _totalMoney - totalPay;
41  }
42
43  // 查詢每輪狀態
44  function roundQuery(uint256 index) public returns(RoundStatus memory){
45      require(index < totalHistoryPlaytime);
46      return roundStatus[index];
47  }
48
49  // 查詢提領紀錄
50  function charityQuery(address addr) public returns(uint256[] memory){
```

```
51     return charityRecord[addr];
52 }
```

執行結果

withdraw(以自己):

 [vm] from: 0x5B3...eddc4 to: Lottery.withDraw(address) 0xd91...39138
value: 0 wei data: 0x0a6...eddc4 logs: 0 hash: 0x902...604da Debug 

status	true Transaction mined and execution succeed
transaction hash	0x902a4c4acea4e4912c784028894f3d910ab87a94f5734d175c3d95452a5604da 
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 
to	Lottery.withDraw(address) 0xd9145CCE52D386f254917e481eB44e9943F39138 
gas	75016 gas 
transaction cost	60431 gas 
execution cost	60431 gas 
input	0x0a6...eddc4 
decoded input	{ "address addr": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4" } 
decoded output	{ } 
logs	[]  
val	0 wei 

歷史開獎次數:

call to Lottery.totalHistoryPlaytime

CALL	[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Lottery.totalHistoryPlaytime() data: 0xef1...ffe22			Debug	^
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4				
to	Lottery.totalHistoryPlaytime()				
	0xd9145CCE52D386f254917e481eB44e9943F39138				
execution cost	23603 gas (Cost only applies when called by a contract)				
input	0xef1...ffe22				
decoded input	{}				
decoded output	{				
	"0": "uint256: 1"				
	}				
logs	[]				

歷史總金額:

call to Lottery.totalHistoryMoney

CALL	[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Lottery.totalHistoryMoney() data: 0x17c...d222b			Debug	^
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4				
to	Lottery.totalHistoryMoney()				
	0xd9145CCE52D386f254917e481eB44e9943F39138				
execution cost	23538 gas (Cost only applies when called by a contract)				
input	0x17c...d222b				
decoded input	{}				
decoded output	{				
	"0": "uint256: 2006"				
	}				
logs	[]				

歷史總玩家:

CALL

[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to: Lottery.totalHistoryPlayer() data: 0x873...6289c

Debug

^

from

0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to

Lottery.totalHistoryPlayer()
0xd9145CCE52D386f254917e481eB44e9943F39138

execution cost

23493 gas (Cost only applies when called by a contract)

input

0x873...6289c

decoded input

{}

decoded output

{
 "0": "uint256: 2"
}

logs

[]

查詢歷屆資訊:

✓

[vm] from: 0x5B3...eddC4
to: Lottery.roundQuery(uint256) 0xd91...39138 value: 0 wei
data: 0x908...00000 logs: 0 hash: 0xa68...17678

Debug

^

status

true Transaction mined and execution succeed

transaction hash

0xa6810c2fdc064c1e380ac8b2496b8eeb778618cb038e7dad26b0dc77bc17678

from

0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

to

Lottery.roundQuery(uint256)
0xd9145CCE52D386f254917e481eB44e9943F39138

gas

37963 gas

transaction cost

33011 gas

execution cost

33011 gas

input

0x908...00000

decoded input

{
 "uint256 index": "0"
}

decoded output

{
 "0": "tuple(address,uint256,uint256):
 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4,1805,2"
}

logs

[]

查詢慈善機構提領紀錄(以自己):

✓

[vm] from: 0x5B3...eddc4

to: Lottery.charityQuery(address) 0xd91...39138 value: 0 wei

data: 0x5c9...eddc4 logs: 0 hash: 0xaa2...1ee47

Debug

▲

status

true Transaction mined and execution succeed

transaction hash

0xaa2af802d48aad07de2e996f667d76fc7066ef19c8d9e330bf507cb69291ee47

📄

from

0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

📄

to

Lottery.charityQuery(address)

0xd9145CCE52D386f254917e481eB44e9943F39138

📄

gas

31360 gas

📄

transaction cost

27269 gas

📄

execution cost

27269 gas

📄

input

0x5c9...eddc4

📄

decoded input

{
 "address addr":
 "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4"
}

📄

decoded output

{
 "0": "uint256[]: 201"
}

📄

logs

[]

📄

📄

val

0 wei

📄