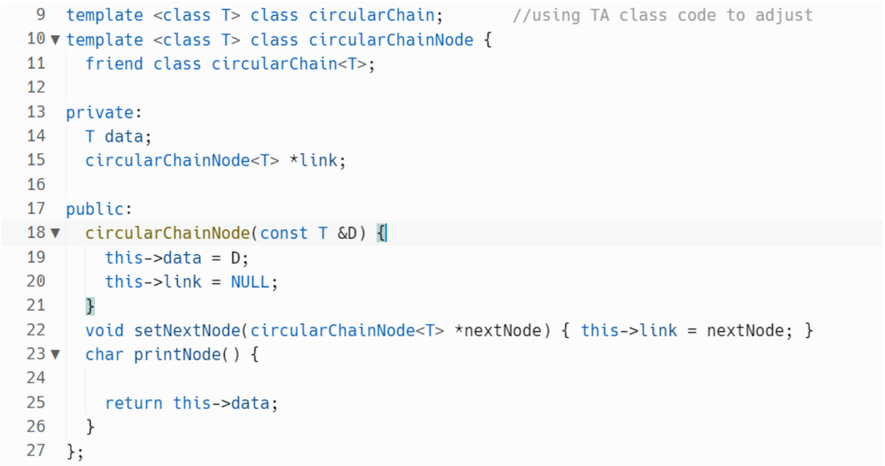
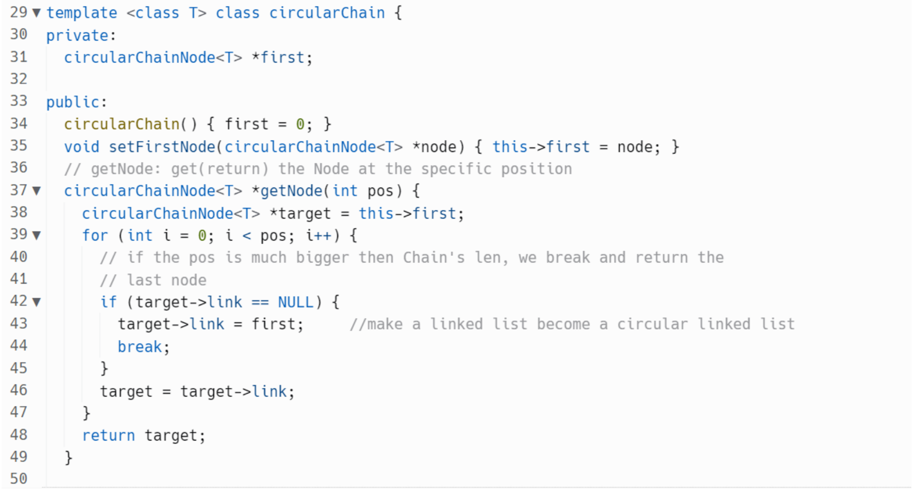
# Data Structure Assignment #2

## 程式相關報告

工資系 113 H54094015 張柏駿

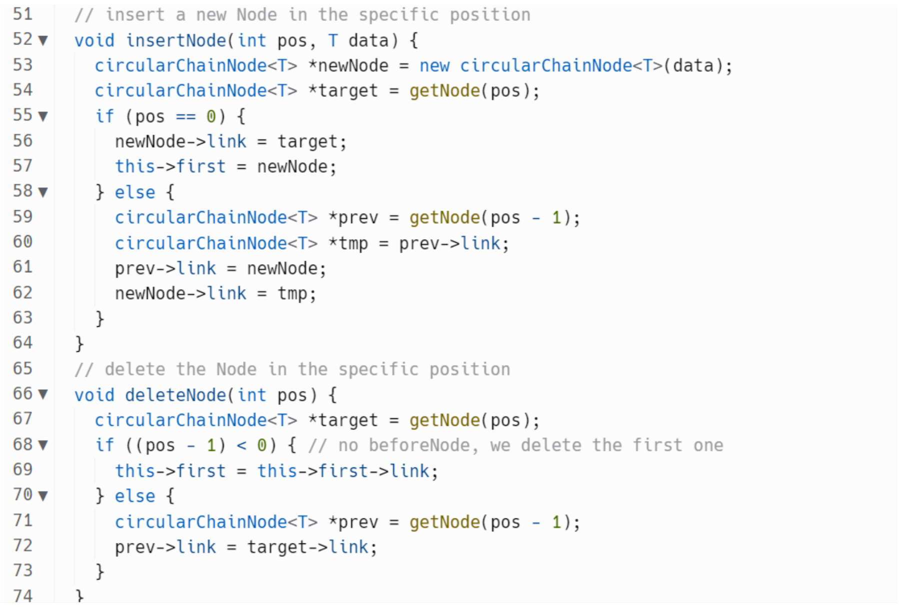


```
template <class T> class circularChain;       //using TA class code to adjust
template <class T> class circularChainNode {
  friend class circularChain<T>;

private:
  T data;
  circularChainNode<T> *link;

public:
  circularChainNode(const T &D) {
    this->data = D;
    this->link = NULL;
  }
  void setNextNode(circularChainNode<T> *nextNode) { this->link = nextNode; }
  char printNode() {

    return this->data;
  }
};
```



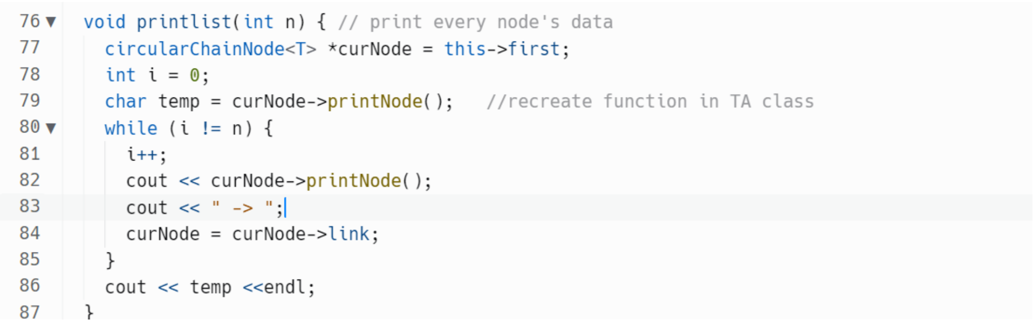
```
template <class T> class circularChain {
private:
  circularChainNode<T> *first;

public:
  circularChain() { first = 0; }
  void setFirstNode(circularChainNode<T> *node) { this->first = node; }
  // getNode: get(return) the Node at the specific position
  circularChainNode<T> *getNode(int pos) {
    circularChainNode<T> *target = this->first;
    for (int i = 0; i < pos; i++) {
      // if the pos is much bigger then Chain's len, we break and return the
      // last node
      if (target->link == NULL) {
        target->link = first;     //make a linked list become a circular linked list
        break;
      }
      target = target->link;
    }
    return target;
  }

```

前面完全使用助教課的程式，改動的地方在這邊將原本的在 linked list 中使用的 getNode 做更改，若在 for 迴圈 traversal 的過程中，當走到最後一個 node(target )時，會將其下一個的指標指向最一開始(頭)的 first，達到 circular 的效果。



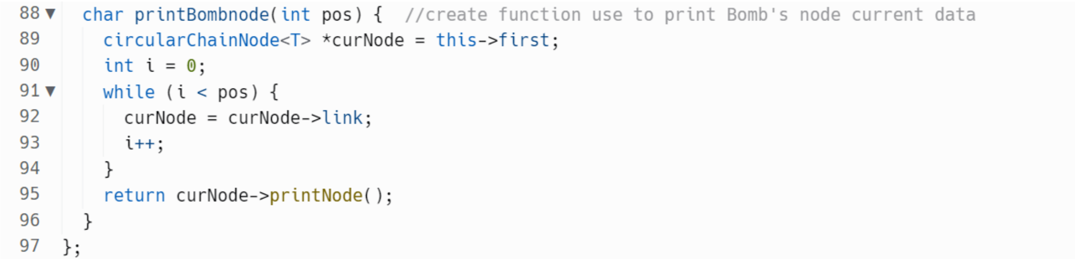
```
// insert a new Node in the specific position
void insertNode(int pos, T data) {
  circularChainNode<T> *newNode = new circularChainNode<T>(data);
  circularChainNode<T> *target = getNode(pos);
  if (pos == 0) {
    newNode->link = target;
    this->first = newNode;
  } else {
    circularChainNode<T> *prev = getNode(pos - 1);
    circularChainNode<T> *tmp = prev->link;
    prev->link = newNode;
    newNode->link = tmp;
  }
}
// delete the Node in the specific position
void deleteNode(int pos) {
  circularChainNode<T> *target = getNode(pos);
  if ((pos - 1) < 0) { // no beforeNode, we delete the first one
    this->first = this->first->link;
  } else {
    circularChainNode<T> *prev = getNode(pos - 1);
    prev->link = target->link;
  }
}
```

都使用助教課的 function，沒有做更動。



```
void printlist(int n) { // print every node's data
  circularChainNode<T> *curNode = this->first;
  int i = 0;
  char temp = curNode->printNode();   //recreate function in TA class
  while (i != n) {
    i++;
    cout << curNode->printNode();
    cout << " -> ";
    curNode = curNode->link;
  }
  cout << temp <<endl;
}
```

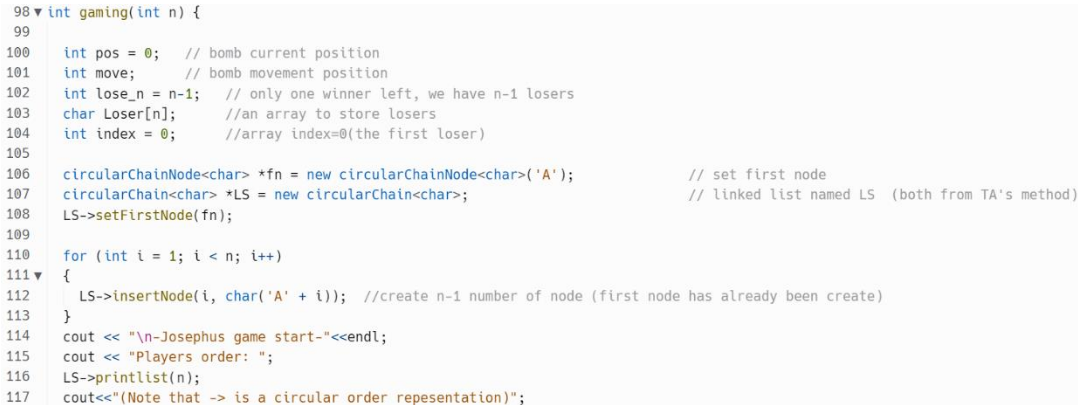
這邊改寫助教課的 printlist() function，其引數為當前遊戲人數，為了在輸出時達到如 A->B->A 的效果，多設定了一個 temp 變數，可以在每次 printlist()時，在還沒進入 while()迴圈時記錄下第一個 node 的 Data，最後在迴圈外輸出就可以在終端機螢幕上同樣顯示出 circular 的效果。



```
  char printBombnode(int pos) {  //create function use to print Bomb's node current data
    circularChainNode<T> *curNode = this->first;
    int i = 0;
    while (i < pos) {
      curNode = curNode->link;
      i++;
    }
    return curNode->printNode();
  }
};
```

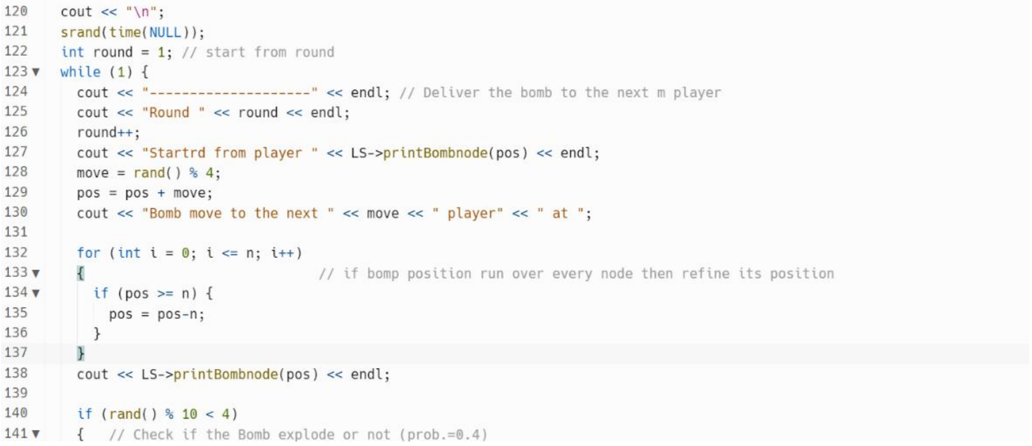
這個 printBombnode() function ，其引數為炸彈位置，可以印出炸彈的 node

位置，之所以可以這樣更動是因為在助教原本的程式中在做 insertnode 時就有設定傳入 pos 的位置，同樣用 while 迴圈進行 traversal，一開始我設定 void 的函數，裡面沒有回傳值(即使用 cout)，但後續要將爆炸的玩家資料存入輸家的陣列中會出現問題，詢問同學後改成有回傳值的函式因此將回傳值型態改為 char。



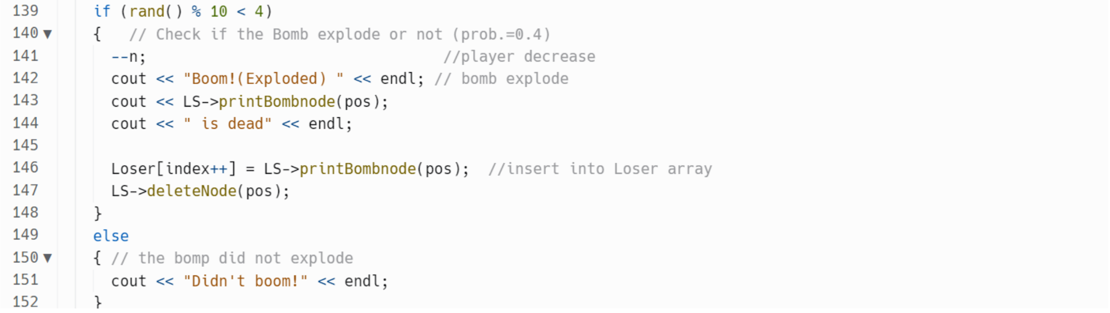
```
int gaming(int n) {

  int pos = 0;    // bomb current position
  int move;       // bomb movement position
  int lose_n = n-1;   // only one winner left, we have n-1 losers
  char Loser[n];      //an array to store losers
  int index = 0;      //array index=0(the first loser)

  circularChainNode<char> *fn = new circularChainNode<char>('A');             // set first node
  circularChain<char> *LS = new circularChain<char>;                          // linked list named LS  (both from TA's method)
  LS->setFirstNode(fn);

  for (int i = 1; i < n; i++)
  {
    LS->insertNode(i, char('A' + i));  //create n-1 number of node (first node has already been create)
  }
  cout << "\n-Josephus game start-"<<endl;
  cout << "Players order: ";
  LS->printlist(n);
  cout<<"(Note that -> is a circular order repesentation)";
```

新增一個 gaming() 將作業要求遊戲依序執行的步驟放入，其引數為一 int n，即為要求玩家輸入的玩家數量，宣告等等會用到的包括遊戲輪數，炸彈位置、炸彈步長，以及陣列和陣列索引等等。用助教課和助教一模一樣的 code 來新增最初的 first node(FN)和 Circular linked list (CLS)，用一個迴圈依序插入 node，迴圈可以依序加入 pos 值和 char 的 Data，覺得用 A 的 ASCII 可以很方便的新增 data，之後並宣布遊戲開始，印出當前玩家順序。



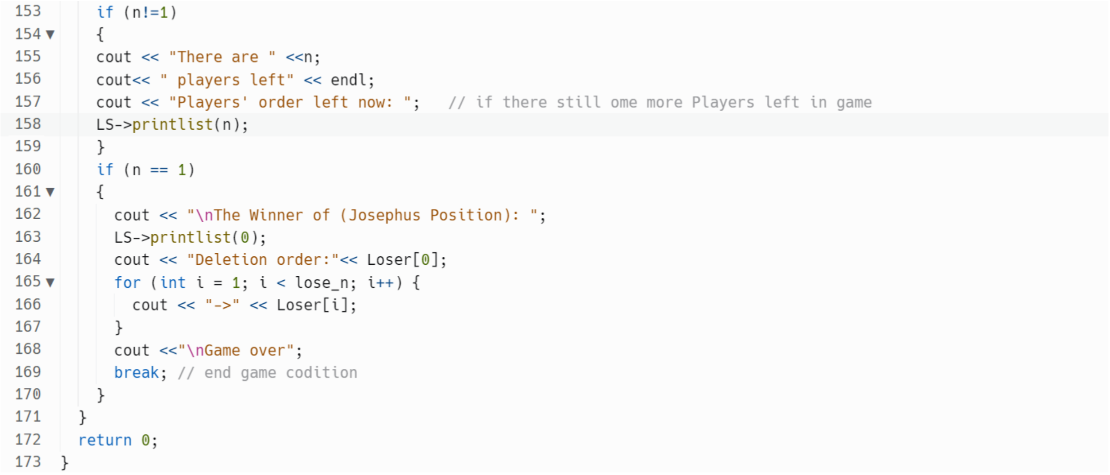
```
  cout << "\n";
  srand(time(NULL));
  int round = 1; // start from round
  while (1) {
    cout << "--------------------" << endl; // Deliver the bomb to the next m player
    cout << "Round " << round << endl;
    round++;
    cout << "Startrd from player " << LS->printBombnode(pos) << endl;
    move = rand() % 4;
    pos = pos + move;
    cout << "Bomb move to the next " << move << " player" << " at ";

    for (int i = 0; i <= n; i++)
    {                             // if bomp position run over every node then refine its position
      if (pos >= n) {
        pos = pos-n;
      }
    }
    cout << LS->printBombnode(pos) << endl;

    if (rand() % 10 < 4)
    {   // Check if the Bomb explode or not (prob.=0.4)
```

宣告回合從 1 開始，一輪一輪進行遊戲，用一連串的-作為分隔，但在位置部分出現問題，跟同學討論之後，多加了一個 for loop，每跑過一次 circular 都用一開始的玩家數重新更新位置，避免位置持續往上加。



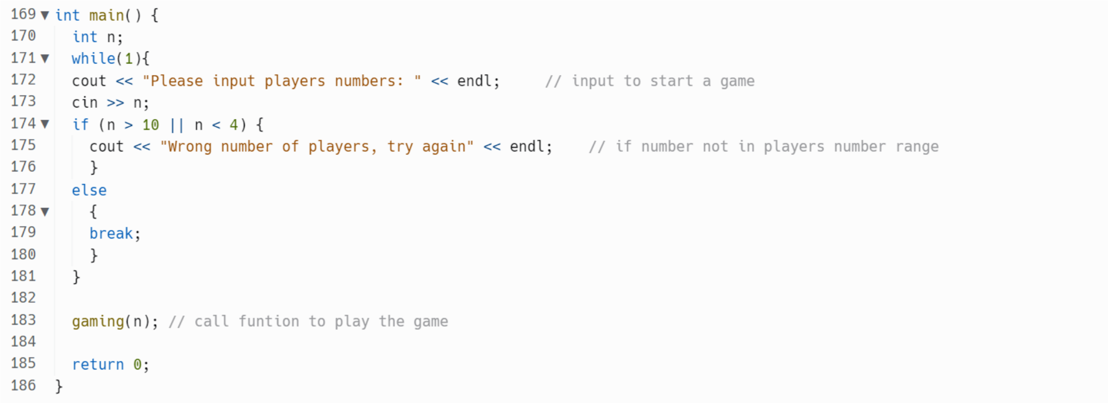
```
    if (rand() % 10 < 4)
    {   // Check if the Bomb explode or not (prob.=0.4)
      --n;                                 //player decrease
      cout << "Boom!(Exploded) " << endl; // bomb explode
      cout << LS->printBombnode(pos);
      cout << " is dead" << endl;

      Loser[index++] = LS->printBombnode(pos);  //insert into Loser array
      LS->deleteNode(pos);
    }
    else
    { // the bomp did not explode
      cout << "Didn't boom!" << endl;
    }
```

依據題目設定爆炸的機率，每回合結束後炸彈有 0.4 的機率會爆炸，若爆炸則用助教 code 裡面的 deleteNode 淘汰該玩家，並指派其 data 進到淘汰者陣列中，且遊戲中的玩家數會減少，下回合則由下個玩家開始傳遞炸彈；若炸彈沒有爆炸，則由接到炸彈的人開始進行下回合的遊戲。



```
    if (n!=1)
    {
    cout << "There are " <<n;
    cout<< " players left" << endl;
    cout << "Players' order left now: ";   // if there still ome more Players left in game
    LS->printlist(n);
    }
    if (n == 1)
    {
      cout << "\nThe Winner of (Josephus Position): ";
      LS->printlist(0);
      cout << "Deletion order:"<< Loser[0];
      for (int i = 1; i < lose_n; i++) {
        cout << "->" << Loser[i];
      }
      cout <<"\nGame over";
      break; // end game codition
    }
  }
  return 0;
}
```

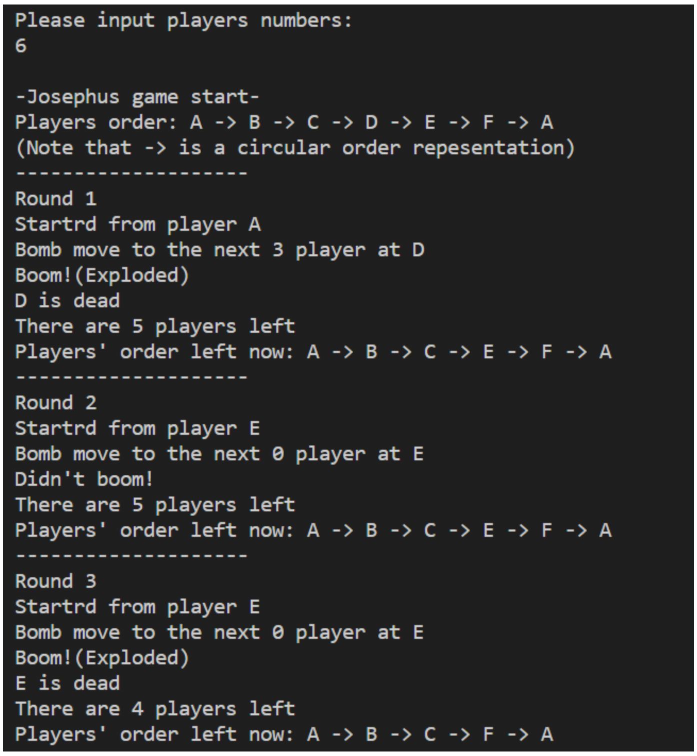
若玩家數不等於 1 人，代表還有兩人以上，遊戲尚未結束，則繼續遊戲，告知有多少剩餘人數在遊戲中，並用 circular 印出其順序，若玩家數 n 剩餘 1 人，則為結束遊戲的情況，依序列印出淘汰者陣列中的資料。



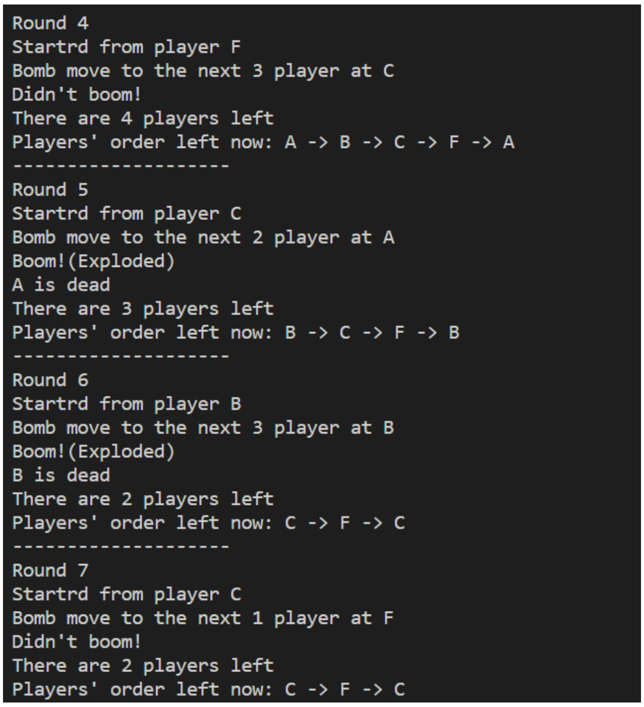
```cpp
int main() {
  int n;
  while(1){
  cout << "Please input players numbers: " << endl;     // input to start a game
  cin >> n;
  if (n > 10 || n < 4) {
    cout << "Wrong number of players, try again" << endl;    // if number not in players number range
    }
  else
    {
    break;
    }
  }

  gaming(n); // call funtion to play the game

  return 0;
}
```

主程式部分，讓使用者輸入欲參加約瑟夫環遊戲的數量，若輸入的數字沒有在4~10 人之間，則重新輸入，若正確則 break 出迴圈，呼叫 gaming()，遊戲開始。

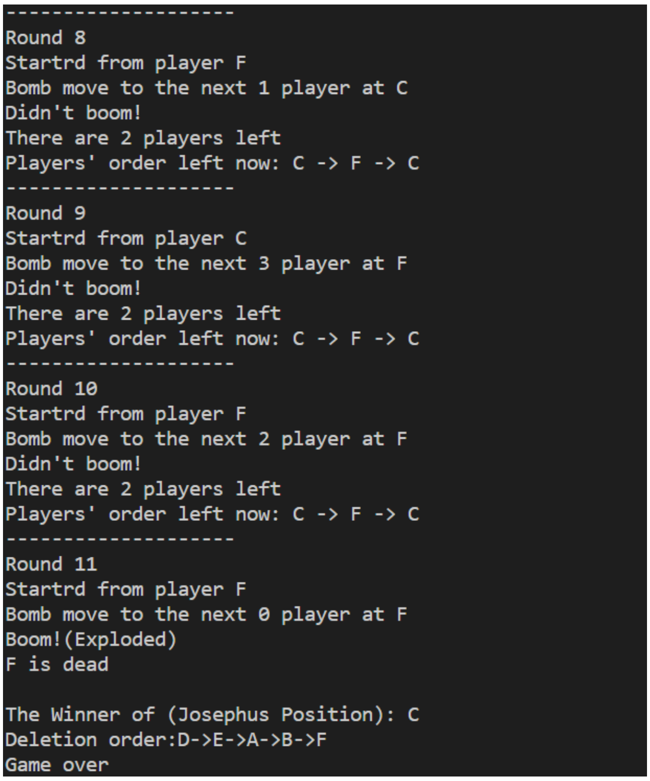
當玩家人數為 6 人時範例輸出結果



```
Please input players numbers:
6

-Josephus game start-
Players order: A -> B -> C -> D -> E -> F -> A
(Note that -> is a circular order repesentation)
--------------------
Round 1
Startrd from player A
Bomb move to the next 3 player at D
Boom!(Exploded)
D is dead
There are 5 players left
Players' order left now: A -> B -> C -> E -> F -> A
--------------------
Round 2
Startrd from player E
Bomb move to the next 0 player at E
Didn't boom!
There are 5 players left
Players' order left now: A -> B -> C -> E -> F -> A
--------------------
Round 3
Startrd from player E
Bomb move to the next 0 player at E
Boom!(Exploded)
E is dead
There are 4 players left
Players' order left now: A -> B -> C -> F -> A
```



```
Round 4
Startrd from player F
Bomb move to the next 3 player at C
Didn't boom!
There are 4 players left
Players' order left now: A -> B -> C -> F -> A
--------------------
Round 5
Startrd from player C
Bomb move to the next 2 player at A
Boom!(Exploded)
A is dead
There are 3 players left
Players' order left now: B -> C -> F -> B
--------------------
Round 6
Startrd from player B
Bomb move to the next 3 player at B
Boom!(Exploded)
B is dead
There are 2 players left
Players' order left now: C -> F -> C
--------------------
Round 7
Startrd from player C
Bomb move to the next 1 player at F
Didn't boom!
There are 2 players left
Players' order left now: C -> F -> C
```



```
--------------------
Round 8
Startrd from player F
Bomb move to the next 1 player at C
Didn't boom!
There are 2 players left
Players' order left now: C -> F -> C
--------------------
Round 9
Startrd from player C
Bomb move to the next 3 player at F
Didn't boom!
There are 2 players left
Players' order left now: C -> F -> C
--------------------
Round 10
Startrd from player F
Bomb move to the next 2 player at F
Didn't boom!
There are 2 players left
Players' order left now: C -> F -> C
--------------------
Round 11
Startrd from player F
Bomb move to the next 0 player at F
Boom!(Exploded)
F is dead

The Winner of (Josephus Position): C
Deletion order:D->E->A->B->F
Game over
```