```cpp
1#include "libBattle.h"
#include <iostream>

using namespace std;
using namespace BattleSpace;

int main()
{
    srand(time(0));                     //Seeds the random number generator
    bool blnContinue=true;              //Flag that determines when to stop the main loop
    bool blnWon = false;                //Flag that keeps track of the state of winning the game
    char chInput = '\0';                //Variable to store user input
    int intIndex = 0;                   //The index of the pointer
    int arrGame[ITEMS];                 //The fixed size array of ones and zeros
    InitArray(arrGame);                 //Initialise the one-dimensional array
    intIndex = InitStart();             //Initiliases the starting index of the pointer

    //Main loop
    do
    {
        system("cls");
        //Output the screen.
        OutputBoard(arrGame,intIndex);
        OutputMenu();
        //Getting user input
        cin >> chInput;
        //Handle movement
        switch(chInput)
        {
        case 'a':
        case 'A':
            Move(arrGame,MOVE_LEFT,intIndex);
            break;
        case 'd':
        case 'D':
            Move(arrGame,MOVE_RIGHT,intIndex);
            break;
        case 'q':
        case 'Q':
            blnContinue = false;
            break;
        default:
            cerr << "Please select a valid option" << endl;
            Pause();
        }
        //Determine if the game has been won
        blnWon = CheckWin(arrGame);
        if(blnWon)
            blnContinue = false;

    }while(blnContinue);

    if(blnWon)
    {
        cout << "*****************************************************************" << endl;
        cout << "*           Congratulations you won the game                    *" << endl;
        cout << "*****************************************************************" << endl;
    }
    else
    {
        cout << "*****************************************************************" << endl;
        cout << "*           Really?  You chose to quite the game?               *" << endl;
        cout << "*****************************************************************" << endl;
    }

    return 0;
}
```

```
#ifndef LIBBATTLE_H_INCLUDED
#define LIBBATTLE_H_INCLUDED

#include <iostream>
#include <cstdlib>
#include <ctime>

const int ITEMS = 20;
const int MOVE_LEFT = 1;
const int MOVE_RIGHT = 2;


namespace BattleSpace
{
    int GetRandom(int intLow, int intHigh);             //Returns a random number, given a lower bound and upper bound number
    void OutputMenu();                                  //Outputs the menu of the user
    void InitArray(int arrNums[]);                      //Initialises the game board
    int InitStart();                                    //Returns the random location of the starting pointer
    void OutputBoard(int arrNums[],int intIndex);       //Outputs the board and the pointer
    void Pause();                                       //Pauses the game
    void Move(int arrNums[],int intMove, int& intIndex);//Moves the pointer and changes the values in the array
    bool CheckWin(int arrNums[]);                       //Determine if the game has been won
}

#endif // LIBBATTLE_H_INCLUDED

```

```cpp
#include "libBattle.h"
using namespace std;

namespace BattleSpace
{
    //Generates a random number.  Used to initialise the gameboard and find starting index
    int GetRandom(int intLow, int intHigh)
    {
        int intRange = intHigh -intLow +1;
        return rand()%intRange+intLow;
    }

    //Outputs only the menu
    void OutputMenu()
    {
        cout << "Change all the values values to the same value:" << endl
             << "A: Move left" << endl
             << "D: Move right" << endl
             << "Q: Quit" << endl;
    }

    //Initialises each array item to either zero or one
    void InitArray(int arrNums[])
    {
        for(int n=0;n<ITEMS;n++)
        {
            arrNums[n] = GetRandom(0,1);
        }
    }

    //Returns the starting location of the index
    int InitStart()
    {
        return GetRandom(0,ITEMS-1);
    }

    //Outputs the one dimensional array and the pointer
    void OutputBoard(int arrNums[],int intIndex)
    {
        for(int n=0;n<ITEMS;n++)
        {
            cout << arrNums[n];
        }
        cout << endl;
        //Output a number of spaces so that the ^ is under the correct number.
        for(int n=0;n<intIndex;n++)
            cout << " ";
        cout << "^" << endl;
    }

    //Waits for input after a cin instruction
    void Pause()
    {
        cin.ignore(100,'\n');
        cout << "Press Enter to continue" << endl;
        cin.get();
    }

    //Moves the pointer to the left and the right.
    void Move(int arrNums[],int intMove,int& intIndex)
    {
        switch(intMove)
        {
        case MOVE_LEFT:
            if(intIndex!=0)          //Ensures that the pointer was not at index 0
                intIndex--;
            break;
        case MOVE_RIGHT:
            if(intIndex!=ITEMS-1)    //Ensures that the pointer was not at index ITEMS-1
                intIndex++;
            break;
        default:
            cerr << "Move function should never get here" << endl;
            Pause();
        }

        //Flips the value of the item
        if(arrNums[intIndex]==0)
            arrNums[intIndex]=1;
        else
            arrNums[intIndex]=0;
    }

    //Determines if the game was won.
```

```
85      bool CheckWin(int arrGame[])
86      {
87          int intFirst = arrGame[0];
88          //Go through each element.  If all elements are the same as the first element, then all
    values are the same.
89          for(int n=1;n<ITEMS;n++)
90              if(arrGame[n]!=intFirst)
91                  return false;
92
93          return true;
94      }
95  }
96
```