Mohammed Muzammil Aziz 218015370

COMP315 Project Documentation

Game of Thrones Trivia

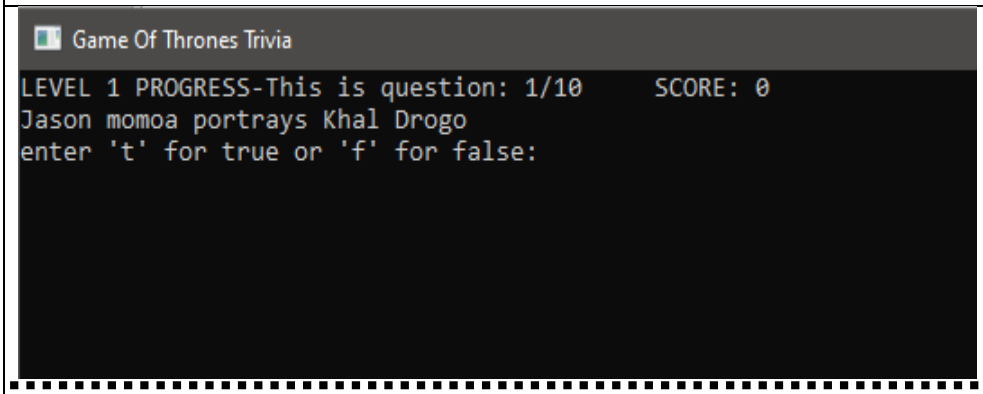
Table of Contents

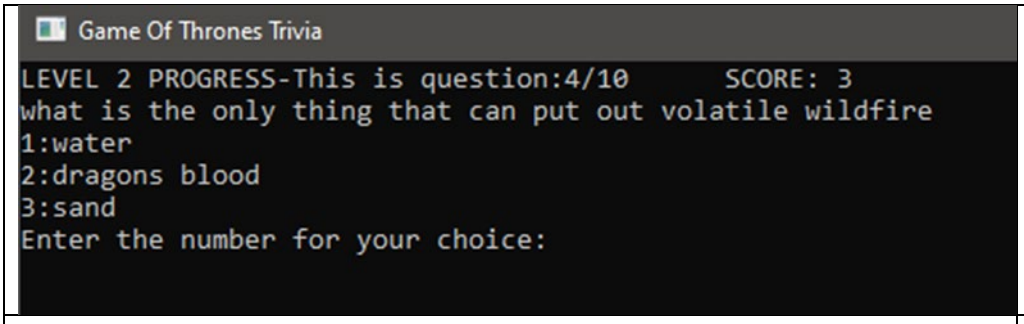
1) Introduction

I have created my console-based quiz game on the world's largest television series, Game of Thrones which is the adaptation of George RR Martin's epic fantasy book series which gives me plenty of resource material to base my quiz on. My game consists of three levels. Each level has 10 questions which is displayed individually and for each question that's answered correctly the player earns a point. The player has to complete each level before proceeding to the next, the user can view his/her progress while playing the game. At the end of the game the user is presented with their final score, they then have the option to post their score to the leader board to see how they did compared to their opponents. This game puts any fan to the test with progressively challenging questions. The game is well laid out and is easy to navigate with no training required to use the program.

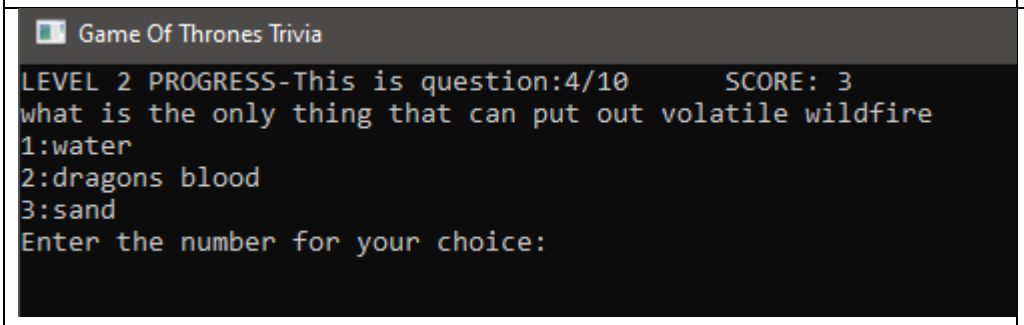
2) Question Structure

1) True/False questions, Multiple-Choice Questions

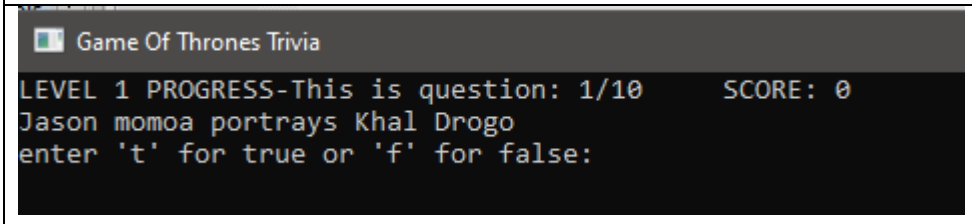
Screenshot	Explanation
	These screenshots are taken from level 1 and level 2. They show that level 1 consists of true and false questions and level 2 consists of multiple choice questions. Level 3 is similar to level 2

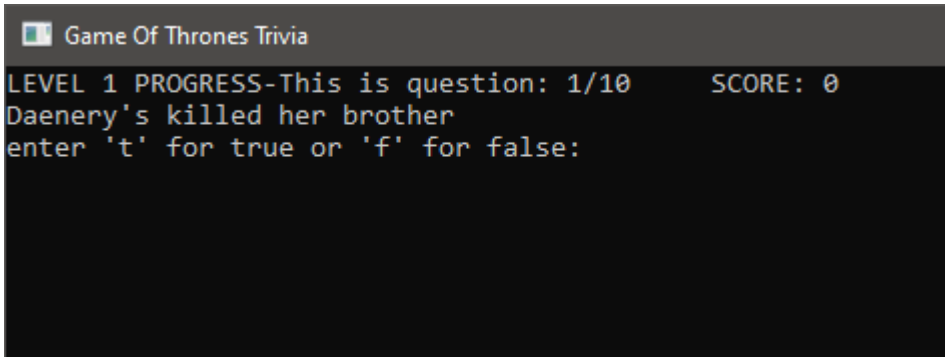
	except the questions become increasingly difficult.

2) Questions appear individually

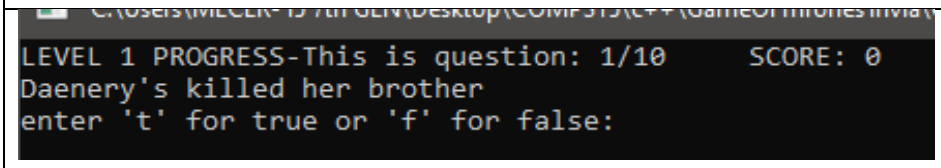
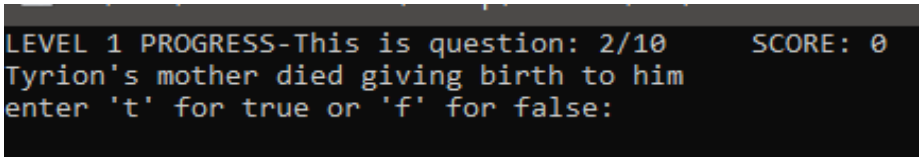
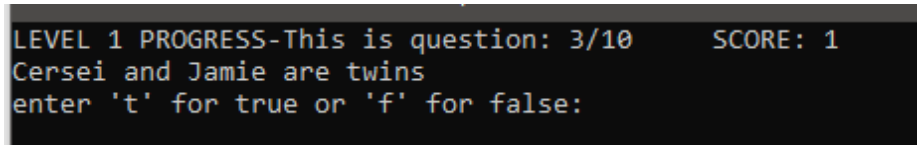
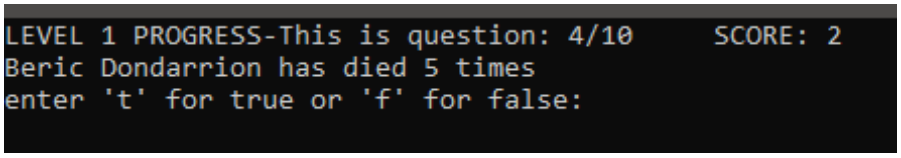
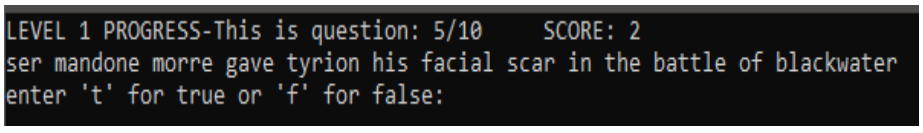
Screenshot	Explanation
	This shows that the player is on question 4. The user is shown the question and his/her choices and is then asked for their choice. Each question is displayed individually in the same way.

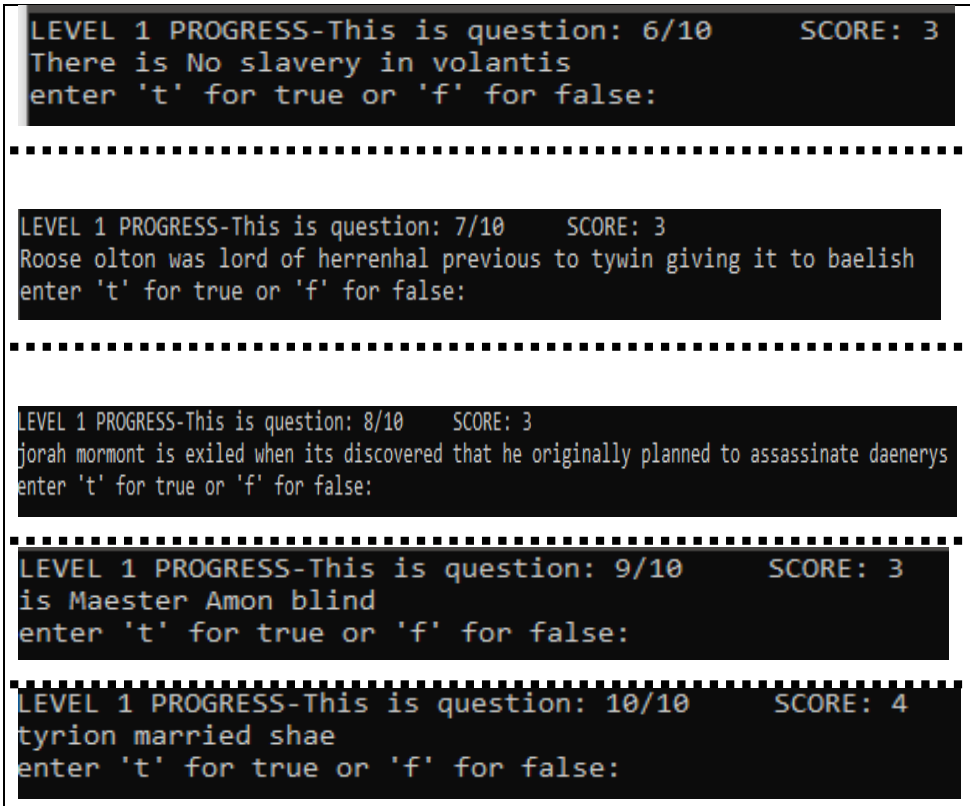
3) Question order is different for every quiz attempt

Screenshot	Explanation
	These screenshots show two different attempts at the quiz. They

	display the quiz info at the top and below that the question. The user is then asked for input

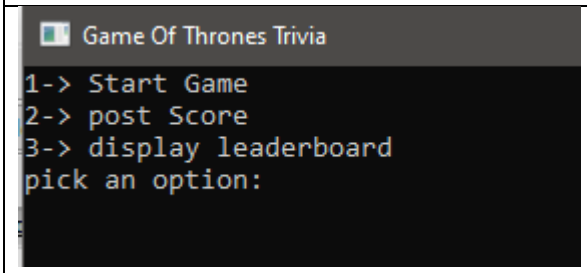
4) Unique question order - no duplicates

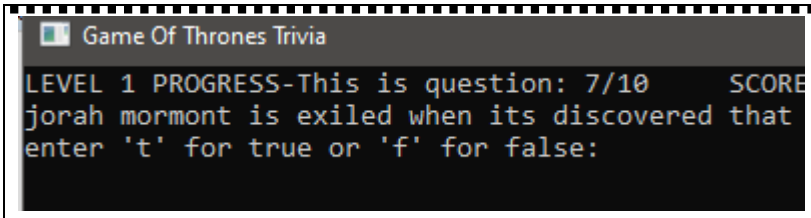
Screenshot	Explanation
    	These screenshots show a level of the game. It shows that each question is unique and that there is no repetition

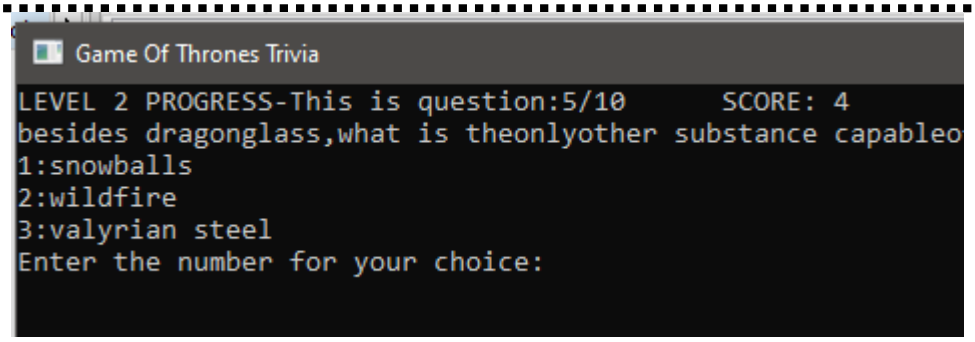
	

3)User Interaction

1) User input

Screenshot	Explanation
	Slide 1-the user navigates the game using the number pad Slide 2- for the true of false questions the user is required to enter “t” or “f”





as their input for the answer Slide 3- for level 2 and 3 the user is required to enter their option using the number pad.

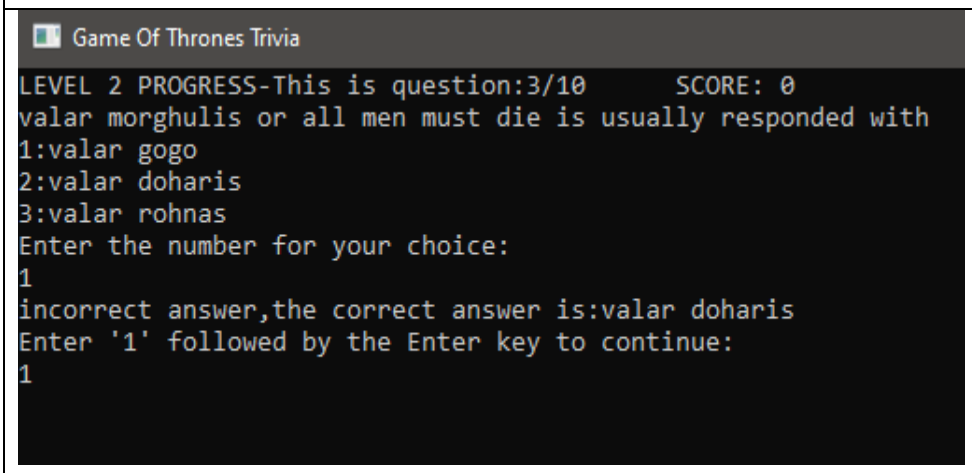
Code Screenshot
Enter options

```
inline void displaypg(int &opt){
    system("CLS");
    cout<<"1-> Start Game"<<endl;
    cout<<"2-> post Score"<<endl;
    cout<<"3-> display leaderboard"<<endl;
    cout<<"pick an option:"<<endl;
    cin>>opt;
}
void selectoptions(int i){
     if(i==1){
        levell ll;
    }else if(i==2){
        string n;
        player playerinfo;
        cout<<"Enter your name:"<<endl;
        cin>>n;
        playerinfo.name=n;
        cout<<playerinfo.name+"this"<<endl;
      writetextfile("26*john");
    }else if(i==3){
       leaderBoard lb;
    }
```

Enter answer:

```
display2(myText);//sends the line of code to the display2 function
cin>>sinput;
int iInput=stoi(sinput);
if(iInput==icorrectanswer2){
    score2=score2+1;
    cout<<"your answer is correct"<<endl;
}else{
    cout<<"incorrect answer,the correct answer is:"<<correctanswer2<<endl;
}//else
```

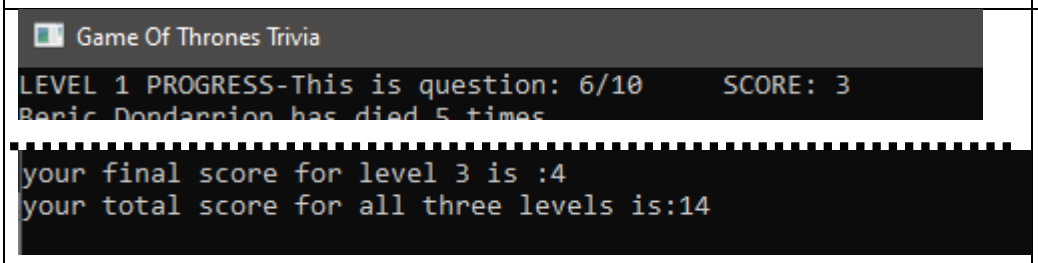
2) User feedback

Screenshot	Explanation
	This screenshot is from level 2. It shows the user has entered an incorrect answer. The game tells the user that their answer is incorrect along with what the correct answer is.

Code Screenshot

```
void displayfeedback2(){
    cout<<"your final score for level 2 is: "<<score2<<endl;
    playerdetails pd;
    pd.settotalscore(score2);
    cout<<"your total score for level 1 and level 2 is: "<<pd.gettotalscore()<<endl;
    cout<<"Enter 1 to continue"<<endl;
    int i;
    cin>>i;
    if(i==1){
        new level3;
    }else{
        cout<<"invalid input"<<endl;
    }
}
```

3) Score accumulator

Screenshot	Explanation
	The player has to answer 10 questions in each level. (screenshot 1)As the user is playing the game the user has the ability to see their score for that level (screenshot 2) At the end of each level the user is shown their total score

	which includes the level they played as well as the previous levels.

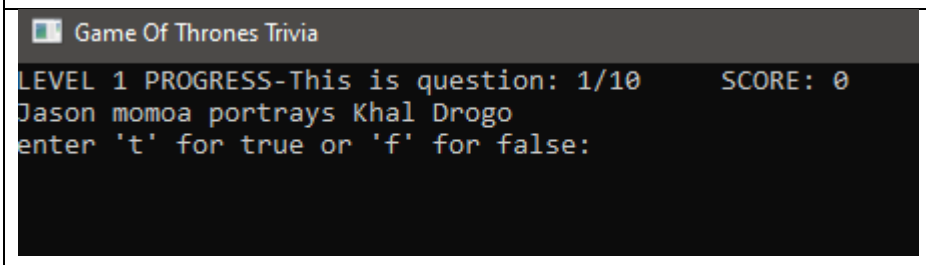
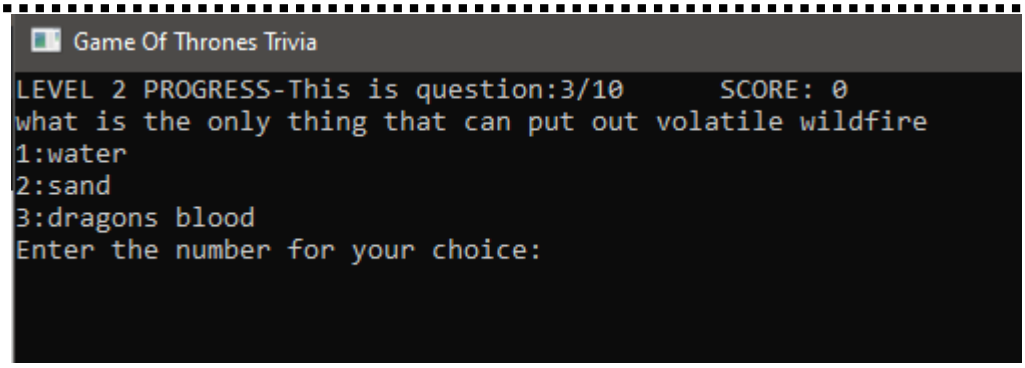
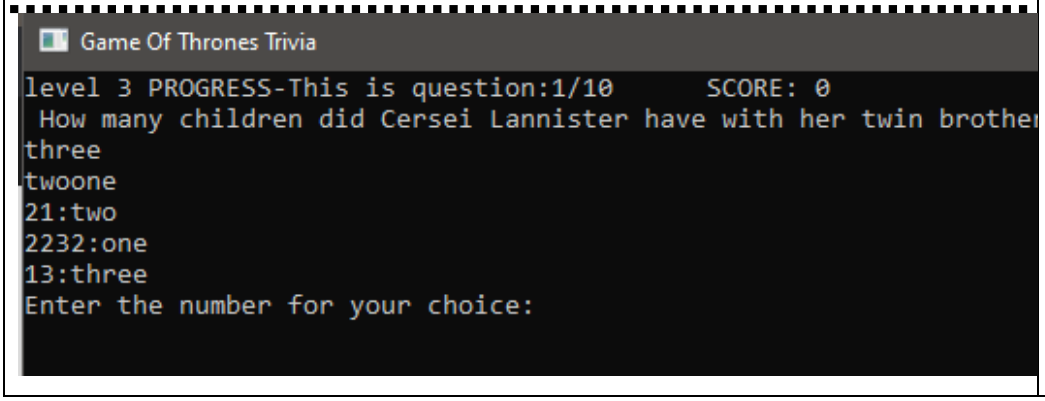
Code Screenshot

```
for(int i=0;i<10;i++){
        if(n[i]==inum){//if the number in the array matches the question number in the textfile
                display2(myText);//sends the line of code to the display2 function
                cin>>sinput;
                int iInput=stoi(sinput);
                if(iInput==icorrectanswer2){
                    score2=score2+1;
                    cout<<"your answer is correct"<<endl;
                }else{
                    cout<<"incorrect answer,the correct answer is:"<<correctanswer2<<endl;
                }//else
                cout<<"Enter '1' followed by the Enter key to continue:"<<endl;
                cin>>sinput;
        }//if
}//for
```

```
void displayfeedback2(){
    system("CLS");//clear the console
    cout<<"your final score for level 2 is: "<<score2<<endl;
    playerdetails pd;
    pd.settotalscore(score2);
    cout<<"your total score for level 1 and level 2 is: "<<pd.gettotalscore()<<endl;
    cout<<"Enter 1 to continue to level 3"<<endl;
    int i;
    cin>>i;
    if(i==1){
        new level3;
    }else{
        cout<<"invalid input"<<endl;
    }
}
```

4) Levels and Progression

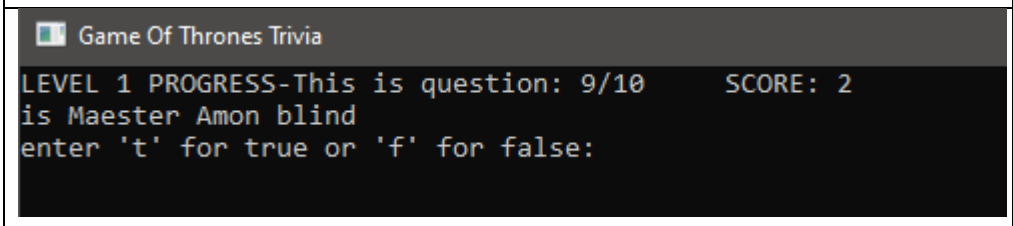
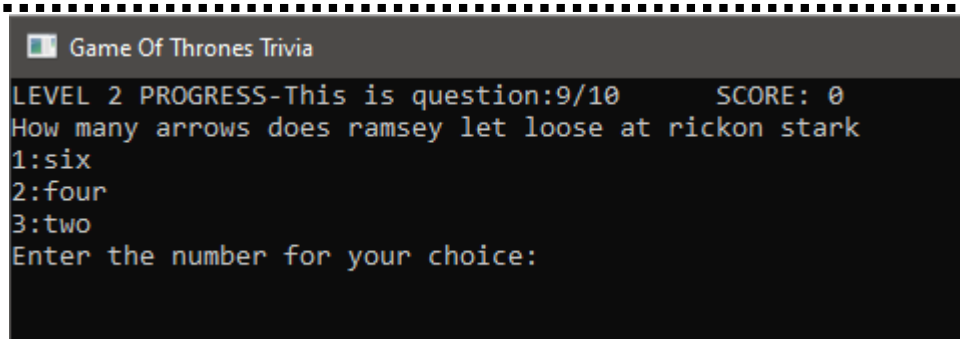
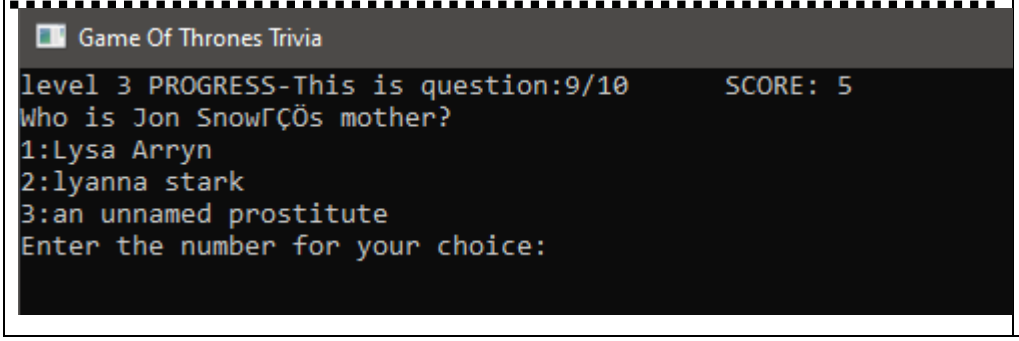
1) Various levels

Screenshot	Explanation
  	These screenshots are taken from each level of the game. The user has to complete each level before proceeding to the next.

Code Screenshot

```
void displayfeedback(){
    system("CLS");//clear the console
    cout<<"your final score for level 1 is: "<<score<<endl;
    playerdetails pd;
    pd.settotalscore(score);
    cout<<"your total score for the game this far is: "<<pd.gettotalscore()<<endl;
    cout<<"Enter 1 to continue to level 2"<<endl;
    int i;
    cin>>i;
    if(i==1){
        new level2;
    }else{
        cout<<"invalid input"<<endl;
    }
}
```

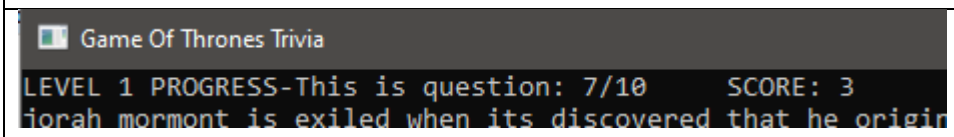
2) At least three levels and eight questions each

Screenshot	Explanation
  	These screenshots show each level where the player is on question 9 of 10.the player has to answer 10 questions in each level

Code Screenshot

```
while (getline (MyReadFile, myText)) {
  // Output the text from the file
  int j=myText.find_first_of("*");
  string sNum=myText.substr(0,j);
  int inum=stoi(sNum);
  for(int i=0;i<10;i++){
          if(n[i]==inum){//if the number in the array matches the question number in the textfile
                  display2(myText);//sends the line of code to the display2 function
                  cin>>sinput;
                  int iInput=stoi(sinput);
                  if(iInput==icorrectanswer2){
                      score2=score2+1;
                      cout<<"your answer is correct"<<endl;
                  }else{
                      cout<<"incorrect answer,the correct answer is:"<<correctanswer2<<endl;
                  }//else
                  cout<<"Enter '1' followed by the Enter key to continue:"<<endl;
                  cin>>sinput;
          }//if
  }//for
  //cout<<inum<<endl;
  //cout << myText<<endl;
}//while
// Close the file
MyReadFile.close();
```

3) Display progress of the quiz

Screenshot	Explanation
	This screenshot shows the level the user is on along with how many questions they have completed

Code Screenshot

```
void display(string s){
     system("CLS");//clear the console
     questionCounter++;
     cout<<"LEVEL 1 PROGRESS-This is question: "<<questionCounter<<"/10      ";
     cout<<"SCORE: "<<score<<endl;
```

5) Programming Techniques

1) Function

Screenshot:

```
void generateRandomNumbers2(){
  int numbers[10]={0,0,0,0,0,0,0,0,0,0};//array to store random question numbers with default values
  srand(time(0));
  int rNumber;
  int i=0;
  bool bfound;
  while (i<10){
     rNumber=1+(rand()%20);
     bfound=false;
     for(int j=0;j<10;j++){
         if(numbers[j]==rNumber){
             bfound=true;
         }//if
     } //for
     if(bfound==false){
         numbers[i]=rNumber;
         //cout<<numbers[i]<<endl;
         i++;
     }
  }//while
  readTFile2(numbers);
}//generate Numbers
```

```
void readTFile2(int *n){
     string sinput;//used to get user input
 // Create a text string, which is used to output the text file
     string myText;
 // Read from the text file
    ifstream MyReadFile("qlevel2.txt");
  // Use a while loop together with the getline() function to read the file line by line
   while (getline (MyReadFile, myText)) {
     // Output the text from the file
     int j=myText.find_first_of("*");
     string sNum=myText.substr(0,j);
     int inum=stoi(sNum);
     for(int i=0;i<10;i++){
             if(n[i]==inum){//if the number in the array matches the question number in the textfile
                     display2(myText);//sends the line of code to the display2 function
                     cin>>sinput;
                     int iInput=stoi(sinput);
                     if(iInput==icorrectanswer2){
                         score2=score2+1;
                         cout<<"your answer is correct"<<endl;
                     }else{
                         cout<<"incorrect answer,the correct answer is:"<<correctanswer2<<endl;
                     }//else
                     cout<<"Enter '1' followed by the Enter key to continue:"<<endl;
                     cin>>sinput;
```

```
void display2(string s){
    system("CLS");//clear the console
    questionCounter2++;
    cout<<"LEVEL 2 PROGRESS-This is question:"<<questionCounter2<<"/10      ";
    cout<<"SCORE: "<<score2<<endl;
    string incorrectans1;
    string incorrectans2;
    int j=s.find_first_of("*");
    string temp;//used to display up until the delimiter
    temp=s.substr(0,j);
    //display the question
    s=s.substr(j+1,s.length());
    j=s.find_first_of("*");
    temp=s.substr(0,j);
    cout<<temp<<endl;
    //get the correct answer from text file
    s=s.substr(j+1,s.length());
    j=s.find_first_of("*");
    correctanswer2=s.substr(0,j);//stores the correct answer in correctanswer2
    //get the remaining incorrect answers
    s=s.substr(j+1,s.length());
    j=s.find_first_of("*");
    incorrectans1=s.substr(0,j);
    s=s.substr(j+1,s.length());
    j=s.find_first_of("*");
    incorrectans2=s.substr(0,j);
    shuffle2(correctanswer2,incorrectans1,incorrectans2);
```

```
void shuffle2(string s1,string s2,string s3){
    int answerpositions[3]={0,0,0};//stores the order in which the answers will be displayed
     int randoma;
     int n=0;
     while(n<3){
        randoma=1+(rand()%3);
        bool bfound=false;
        for(int j=0;j<3;j++){
              if(answerpositions[j]==randoma){
                  bfound=true;
              }//if
        } //for
        if(bfound==false){
             if(randoma==1){
                 icorrectanswer2=n+1;
                 cout<<n+1<<":"<<s1<<endl;
             }else if(randoma==2){
                 cout<<n+1<<":"<<s2<<endl;
             }else if(randoma==3){
                 cout<<n+1<<":"<<s3<<endl;
             }//else
             answerpositions[n]=randoma;
             n++;
        }//if bfound==false
     }//while
} //shuffle
```

Motivation:

Functions are used within each class to break a large task into smaller tasks. For example to display the questions for the game. First the function generateRandomNumbers2 must generate 10 unique random numbers between 1 and 20 and store these numbers in an array. The readTFile2 function uses a pointer to access to array. The readTFile2 class then checks if a random number that was generated matches a question number in the text file. If it does then that line of the text file (question number*question*correct answer*incorrect answer*incorrect answer) is sent to the display2 function. The display2 function then break

that line from the text file into its individual components. It displays the question and the three answers then get sent to the shuffle2 function, the answers get shuffled and the correct answer along with the option gets stored in global variable to be checked after the user input.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	Was used to break a large task into smaller tasks, making the code more modular.

2) Class

Screenshot:

main.cpp | leaderBoard.cpp | leaderBoard.h | playGame.cpp | playGame.h | level1.cpp | level1.h | level2.cpp | level3.cpp | playerdetails.h | playerdetails.cpp

```
int questionCounter;
level1::level1()
{
    score=0;
    generateRandomNumbers();
    displayfeedback();
}
void generateRandomNumbers(){
 int numbers[10]={0,0,0,0,0,0,0,0,0,0};//array to store random question numbers with default values
 srand(time(0));
 int rNumber;
```

Motivation:

Classes allowed me to separate each level of the game, which simplified the programming. Each class displays one level of the game, this also allowed for each level to be different in the way the questions are displayed as well as the way the user interacts with the game.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective

Not met		**completely, provide a short explanation to support the claim**
Partially		
Completely	x	Each part of the game has its own class. This has allowed my program to be modular.

3) Struct

Screenshot:

```
struct player{
    string name;
    int scoretotal;
};
```

Motivation:

Allows the program to access the player name and score from any function that follows the structure definition.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	Lets more than one function use the same variables.

4) Pointer

Screenshot:

```
void readTFile3(int *n){//receives a pointer to the array as a parameter
    string sinput;//used to get user input
// Create a text string, which is used to output the text file
    string myText;
```

```
readTFile3(numbers);
```

Motivation:

The readTfFile function receives the memory address of the array (which stores 10 random numbers between 1 and 20) as a pointer. The function then uses this pointer to match the numbers in the array with the one in the text file. The pointer means that I do not need to have multiple copies of the array which makes my program more efficient.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	Allows me to refer to another object without making of copy of it.

5) Reference

Screenshot:

```
void sortvector(std::vector<string> &data1);
void displayplayers(std::vector<string> &data1);
```

Motivation:

Referencing allows me to use a vector that's declared in a different function in this function. By using a reference, it allows me to use the vector without having to create a copy of it which makes the program more efficient.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	Allows me to refer to another object without making of copy of it.

6) Data Structures-Array

Screenshot:

```
int numbers[10]={0,0,0,0,0,0,0,0,0,0};//array to store random question numbers with default values
srand(time(0));
**********************************************************************

*******

void generateRandomNumbers(){
  int numbers[10]={0,0,0,0,0,0,0,0,0,0};//array to store random question numbers with default values
  srand(time(0));
  int rNumber;
  int i=0;
  bool bfound;
  while (i<10){
     rNumber=1+(rand()%20);
     bfound=false;
     for(int j=0;j<10;j++){
         if(numbers[j]==rNumber){
             bfound=true;
         }//if
     } //for
     if(bfound==false){
         numbers[i]=rNumber;
         //cout<<numbers[i]<<endl;
         i++;
     }
  }//while
  readTFile(numbers);
}//generate Numbers
```

Motivation:

The program generates 10 unique random numbers between 1 and 20. This gets stored in an array since there has to be 10 numbers generated (one number for each question). The array is then sent to the readTFile function. If a number that was generated in the array matches a question number in the text file, the question is displayed.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	An array was used since each time the code runs. There has to be 10 random questions generated. Since the number of questions that are displayed won't change it makes using an array more

		efficient than a dynamic data structure.

7) Vector

Screenshot:

```
std::vector<string> players;//stores each line from the text file
```

```
void readtextfile(){
    // Create a text string, which is used to output the text file
    string myText;
// Read from the text file
   ifstream MyReadFile("leaderB.txt");
 // Use a while loop together with the getline() function to read the file line by line
  while (getline (MyReadFile, myText)) {
    // Output the text from the file
    players.push_back(myText);
  }//while
 // Close the file
  MyReadFile.close();
}
```

Motivation:

I decided to use a vector to store each line of the text file for the leader board. The reason for using a vector is that it is dynamic. So an infinite number of people can play the game and add their score to the leader board.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	The leader board had to use a vector because it needed to be dynamic since the number of people that will be added to the leader board will increase each time someone plays the game.

8) Class Template

Screenshot:

```
template <class T>
class  swapplayer{
public:
    swapplayer(T &b1,T &b2){
        T temp;
        temp=b2;
        b2=b1;
        b1=temp;
    }
};
```

Motivation:

The class template allows me to swap any two variables. This lets me use one class definition to swap strings or integers. It is used to swap the data in the vector when its being sorted in descending order.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	Allowed me to swap either the player name or the players score using the same class.

9) Operator Overloading

Screenshot:

```
void display3();
void display3(string s);
```

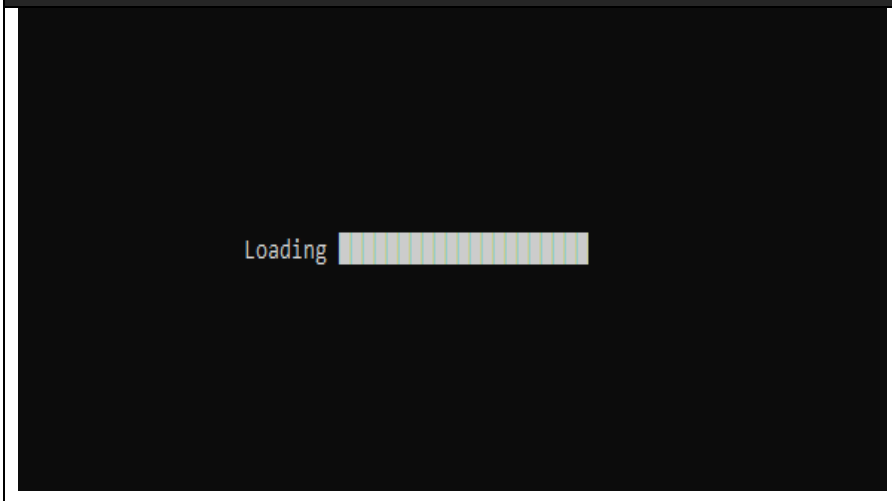
```
display3();//displays the progress and score
display3(myText);//used to seperate the line and display the question and answers
cin>>sinput;
```

Motivation:

Operator overloading allows me to use the same function definition but display different outputs depending on the parameter that's passed to the function.

How have you met the objectives?	Cross (X) the appropriate box	If you think that you have met the objective completely, provide a short explanation to support the claim
Not met		
Partially		
Completely	x	Simplified the program by having one definition for output.

6) Additional Items

Screenshot	What does your quiz include?
	Loading bar-I have added a loading bar at the start of the game. This does not only add a visual aspect to the game but also gives the user a sense of anticipation to play the game.

Code Screenshot

```
cout<<"\n\n\n\n\n\n\n\n\n\n\n\n";
 cout<<"\t\t\t\t\tLoading ";
 char x=219;
 for(int i=0;i<35;i++){
      cout<<x;
      if(i<10){
          Sleep(300);
      }
      if(i>=10 && i<30){
          Sleep(150);
      }
      if(i>=100){
          Sleep(25);
      }
 }
 playGame pg;
```

Screenshot	What does your quiz include?
Game Of Thrones Trivia score name 1- 28 will 2- 27 johnathan 3- 26 david 4- 24 tim 5- 22 ali 6- 16 john 7- 12 smith	Leader board- I have included a leader board in my game to add an extra element of competitiveness to the game. Each user has the option to post their score to the leader board and see how they did compared to other players.

Code Screenshot

```
void readtextfile();
void sortvector(std::vector<string> &data1);
void displayplayers(std::vector<string> &data1);
leaderBoard::leaderBoard()
{
    readtextfile();
    sortvector(players);
    displayplayers(players);
}
void readtextfile(){
```