TEJ4M

Summative Project – Computer Game (Tentative)

For this project, you will create a computer game. It is recommended that you use an existing game as the basis for your program. You may also create your own game, but this option has the added responsibility of (a) the creative process to make a reasonable game, and (b) producing documentation for the game itself.

Games involving dice, cards, letters, and words provide good candidates. The following list provides some examples of games that are suitable for this project.

- Yahtzee
- Hangman / Wheel of Fortune
- Jeopardy
- Mastermind

Some games or concepts will require additional research beyond what you have covered in the course. It is important that you select a game that is within your abilities. Additional candidate games could include

- Battleship
- Tic-Tac-Toe
- Checkers / Chess
- Minesweeper

Program requirements:

1. All code must be clearly documented.
2. All code must be neatly formatted.
3. Logical and efficient use of variables is required.
4. The program should make use of all programming concepts covered in the course.
5. The program should be as crash-proof as possible.
6. There should be a menu system with an option to get Help.
7. Go above and beyond. Do some research. Insert new features not taught in class.

Grading:

COMM /20

- documentation, proposal and user guide/help of this program
- Spelling and grammar
- Style, layout, presentation

APP /60

- the way in which you have integrated and presented the programming concepts that you have learned in this course. (eg. lists/arrays, functions, data types, conditions, selection statements, loops, etc.).

TI /20

- quality/difficulty of your game
- the way in which you have integrated new features into your game i.e. new functions, libraries etc.

Link to Code (TinkerCad):

https://www.tinkercad.com/things/7gc6lhVMeqB-summative-project-computer-game-tej4m/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard

Raw Code (Word Scramble Game):

```
// Vishwa Dave
// 14/04/2025
// Summative Project – Computer Game (TEJ4M)
// Word Scramble Game

//global variables
int highScore = 0;
int score=0;
const int lengthWordBank = 5; //basically number of word options


//Word Banks
        //Category 1 - Countries
String cat1_L1[] = {"egypt", "india", "japan", "nepal", "spain"}; //level 1
String cat1_L2[] = {"brazil", "turkey", "canada", "france", "jordan"}; //level 2
        //Category 2 - Colours
String cat2_L1[] = {"black", "white", "green", "peach", "coral"}; //level 1
String cat2_L2[] = {"yellow", "orange", "indigo", "maroon", "violet"}; //level 2

void setup()
{
  Serial.begin(9600); //initializing the serial monitor

  //Titile screen
  Serial.println("Welcome to Word Scamble!");
  Serial.println("-------------------------");
  bool startGame = true; //starts game and allows for unlimited replays
```

```
while (startGame){
  //resetting all values for game replay
  String userSelection = "";
  score=0;
  bool loss = false;

  //User selection menu (to see instructions, play, or view highscore)
  Serial.println("\nSelect one of the following: \n\t(i) Instructions\n\t(p) Play\n\t(h) High Score");
  while (Serial.available()==0){}
  userSelection = Serial.readString();


        //Error handling user selection input
  while (userSelection!="i" && userSelection!="I"&&
        userSelection!="p" && userSelection!="P" &&
        userSelection!="h" && userSelection!="H"){
    Serial.println("Not a valid input... please enter 'i', 'p', or 'h': ");
    while (Serial.available()==0){}
    userSelection = Serial.readString();
  }

  //If user wants to see instructions for the game
  if (userSelection=="i" || userSelection=="I"){
    Serial.println("Instructions\n----------------");
    Serial.println("The game will begin with a scrambled 5 letter word. You \n"
                "will have unlimited attempts to figure out the word \n"
                "until the timer runs out. You earn 50 points for each \n"
                "second you save on the timer, and it is added to your \n"
                "total score. After each game a highscore is checked and \n"
                "updated.");
    delay(3000); //adds a delay of 3 seconds so user has time to read screen
  }
  else if (userSelection=="h" || userSelection=="H"){
    //if user wants to see current highscore
    Serial.println("High Score\n------------------");
    Serial.println(highScore);
  }
  else{ //assumed that user has chosen only remaining option "play"

    //Categories selection section
    Serial.println("Select one of the following categories:");
    Serial.println("\t(1) Countries\n\t(2) Colours");
    while (Serial.available()==0){}
    int categorySelection = Serial.parseInt();
    //Error handling category selection input
  while (categorySelection<0 || categorySelection>2){
    Serial.println("Not a valid input... please enter 1 or 2");
    while (Serial.available()==0){}
    categorySelection = Serial.parseInt();
  }

  //if user has selected category 1 (countries)
  if (categorySelection==1){
    Serial.println("You have chosen 'Countries'"); //informing of selection
    for (int i=0; i<3; i++){ //starts levels from level 1 (i=0)
      if (i==0){ //following will run during level 1 - category 2
        for (int j=0; j<3; j++){
          if (!playRound(cat1_L1, 15)){ //automatically runs a round for level 1
            loss = true; //if they lost the round this allows game to end
            break; //doesn't play rest of rounds for this level
          }
          else{
                Serial.println("Correct Guess!\n"); //informs user if they guessed correct
          }
        }

      }
      else if (i==1){ //level 2
        for (int j=0; j<3; j++){
          if (!playRound(cat1_L2, 10)){ //word bank has also now moved up
            loss = true;
            break;
          }
          else{
                Serial.println("Correct Guess!\n");
          }
        }
      }
      else{ //assumed final level, level 3
        for (int j=0; j<3; j++){
          if (!playRound(cat1_L2, 5)){
            loss = true;
            break;
          }
          else{
                Serial.println("Correct Guess!\n");
          }
        }
    }

    if (loss) //if they have lost any round
      break; //this ends the play game
  }

} // end of category1 if loop
else if (categorySelection==2){ //if user has selected category 2 (colours)
  Serial.println("You have chosen 'Colours'"); // informs user of their selection
  for (int i=0; i<3; i++){ //allows for changes between levels 1-3
    if (i==0){ //level 1 - category 2
      for (int j=0; j<3; j++){
        if (!playRound(cat2_L1, 15)){
          loss = true;
          break;
        }
        else{
              Serial.println("Correct Guess!\n");
        }
      }

    }
    else if (i==1){ //level 2 - category 2
      for (int j=0; j<3; j++){
        if (!playRound(cat2_L2, 10)){
          loss = true;
          break;
        }
        else{
              Serial.println("Correct Guess!\n");
        }
      }
    }
    else{
      for (int j=0; j<3; j++){ //level 3 - category 2
        if (!playRound(cat2_L2, 5)){
          loss = true;
          break;
        }
        else{
              Serial.println("Correct Guess!\n");
        }
      }
    }
```

```
          if (loss)
            break; //ends play if they have lost any of the rounds
        }

      } // end of category2 if loop

      if (score>highScore){ //checks if a new high score has been achieved
        highScore = score; //updates highscore
        Serial.println("New High Score: "+String(highScore)); //informs user of new high score
      }
      else{
        Serial.println("Score: "+String(score)); //displays score achieved
        Serial.println("High Score: "+String(highScore)); //displays current high score
      }

    } //end of if statement for if "play" was selected



  } //end of while(startGame) loop
} //end of void setup()

void loop()
{

}

            //Method used to run each round
bool playRound(String categoryName[], int timelimit){
  //pass in the name of the word bank used and timelimit based on level

  randomSeed(analogRead(0)); //initializing random numbers
  long startTime = millis(); //starting timer
  String guess=""; //variable to store their guess
  float elapsedTime; //will store time that has passed

  int randomIdx = random(lengthWordBank); //picks a random index from that list
  String actualWord = categoryName[randomIdx]; //stores what the actual word is
  String scrambleWord = rearrangeWord(actualWord); //creates and stores scrambled version of word
  Serial.println("Your scrambled word is "+scrambleWord); //shows the user the scrambled word

  while (guess!=actualWord && elapsedTime<timelimit){
    //runs while the word isn't guessed and the time hasn't exceeded the time limit

            //asking the user for their guess on what the unscrambled word is
            Serial.println("Enter your guess: ");

    while (Serial.available()==0){
      //while waiting for their response the timer checks if they have exceeded rhe time limit
      elapsedTime = (millis() - startTime)/1000.0;
      if (elapsedTime>timelimit){ //if they have exceeded it then they round ends
        Serial.println("You have run out of time!");
        Serial.println("The word was "+actualWord);
        return false; //returns a loss in round
      }
    }
    guess = Serial.readString(); //guess becomes whatever they have inputted

    if (isValidGuess(guess, actualWord)){ //checks if their guess is even valid
      guess.toLowerCase(); //changes their input to all lowercase to accept capital inputs too
      if (guess!=actualWord){ //if they are wrong, it informs the user
            Serial.println("Incorrect...try again...\n");
      }
    }
    else{ //if they have not inputted a valid guess (i.e. using numbers) then informs them
            Serial.println("Invalid entry...try again...\n");
    }

  }
  score += 50*(timelimit-elapsedTime); //calculates and adds points they earned this round
  return true; //ends round as a win
}

            //method responsible for randomly rearranging words
String rearrangeWord(String word){
  String originalWord = word; //storing word so we can make sure the shuffle worked
  randomSeed(analogRead(1));
  int length = word.length(); //storing length of word

  do{
            for (int i=(length-1); i>0; i--){ //loops from the end to the beginning of the word
      int x = random(i+1); //generates a random index from 0 to i (inclusive)
      char temp = word[i]; //temporarily stores letter
      word[i]=word[x]; //replaces that position with the random index letter
      word[x]=temp; //replaces that random index with letter stored in temp
            }
  } while (word==originalWord); //ensures that the word isnt shuffled back to the original

  return word;
}
```

```
      //method responsible for checking user guesses
bool isValidGuess(String guess, String actual){
  int actualLength = actual.length(); //stores length of actual word
  int enteredLength = guess.length(); //stores length of user's guess

  if (actualLength<enteredLength || actualLength>enteredLength)
    //checking if the length is too long or too short (automatically not going to match word)
    return false; //returns guess as invalid

  for (int i = 0; i < enteredLength; i++) { //goes through each character of entry
    if (!isAlpha(guess[i])) { //if the character isn't a letter
      return false; //returns the entry as invalid
    }
  }

  return true; //assuming other checks have been passed and returns entry as valid
}
```