文本

描述已自动生成

***Report***

**Faculty of Applied Information Technology**

**Field of Study: INFORMATION TECHNOLOGY**

**Specialty: Programming**

AIKYN ARYN ,ABYLAI ASSALOV

No. of student’s record book: 65387 W66749

**programming languages**

Supervisor: mgr inż. Marcin Jagieła

**Rzeszów 2024**

***Report: FamilyFeudConsoleApp***

**Introduction**

**The FamilyFeudConsoleApp is a console-based implementation of the popular game show "Family Feud." In this game, players are presented with a series of questions and must guess the most popular answers to earn points. The goal is to accumulate as many points as possible by the end of the game.**

**Structure and Components**

**The application is organized into the following components:**

**Class Program: The main class that contains the entry point for the application.**

**Nested Class Question: Represents a game question, encapsulating the question text and a list of possible answers with associated points.**

**Key Features**

**Sample Questions and Answers:**

**The game includes a predefined list of questions, each with a set of possible answers and corresponding points.**

**Questions are stored in a static list named Questions.**

**Game Loop:**

**The game iterates through each question in the Questions list.**

**For each question, the player is given up to three attempts to guess an answer.**

**Input Handling:**

**Player input is taken from the console.**

**Answers are processed in a case-insensitive manner to match against the predefined answers.**

**Score Calculation:**

**Points are awarded based on the predefined points for correct answers.**

**The total score is accumulated across all questions.**

**End-of-Game Feedback:**

**Based on the total score, players receive feedback on their performance.**

**Custom messages are displayed for different score ranges.**

**Detailed Code Walkthrough**

**Question Class:**

**csharp**

**class Question**

**{**

**public string Text { get; set; }**

**public List<(string Answer, int Points)> Answers { get; set; }**

**}**

**This class holds the question text and a list of tuples, each containing an answer and its associated points.**

**Sample Questions:**

**csharp**

**static List<Question> Questions = new List<Question>**

**{**

**new Question**

**{**

**Text = "Name a fruit you might eat for breakfast.",**

**Answers = new List<(string, int)>**

**{**

**("Banana", 30),**

**("Apple", 25),**

**("Orange", 20),**

**("Grapes", 15),**

**("Strawberry", 10)**

**}**

**},**

**// Additional questions...**

**};**

**Four sample questions are provided, each with a set of answers and points.**

**Main Method:**

**static void Main(string[] args)**

**{**

**Console.WriteLine("Welcome to Family Feud!");**

**int totalScore = 0;**

**const int maxAttempts = 3;**

**// Game loop through questions**

**// Input handling and scoring logic**

**// End-of-game feedback**

**}**

**The main method initializes the game, handles the game loop, processes input, and calculates the total score.**

**Game Loop and Input Handling:**

**csharp**

**foreach (var question in Questions)**

**{**

**Console.WriteLine($"\nQuestion: {question.Text}");**

**bool answeredCorrectly = false;**

**for (int attempt = 0; attempt < maxAttempts; attempt++)**

**{**

**Console.Write("Your answer: ");**

**string answer = Console.ReadLine()?.Trim().ToLower();**

**if (string.IsNullOrWhiteSpace(answer))**

**{**

**Console.WriteLine("Please enter a valid answer.");**

**attempt--; // Allow the user to retry without counting this as an attempt.**

**continue;**

**}**

**var matchingAnswer = question.Answers.FirstOrDefault(a => a.Answer.ToLower() == answer);**

**if (matchingAnswer != default)**

**{**

**Console.WriteLine($"Correct! You earned {matchingAnswer.Points} points.");**

**totalScore += matchingAnswer.Points;**

**answeredCorrectly = true;**

**break;**

**}**

**else**

**{**

**Console.WriteLine("Sorry, that's not on the board. Try again.");**

**}**

**}**

**if (!answeredCorrectly)**

**{**

**Console.WriteLine("You've exhausted your attempts for this question.");**

**}**

**}**

**The game iterates through each question and allows the player up to three attempts to guess an answer.**

**Input is validated and checked against the list of possible answers.**

**Points are awarded for correct answers.**

**End-of-Game Feedback:**

**csharp**

**Console.WriteLine($"\nGame over! Your total score is: {totalScore}");**

**if (totalScore >= 100)**

**{**

**Console.WriteLine("Great job! You really know your stuff!");**

**}**

**else if (totalScore >= 50)**

**{**

**Console.WriteLine("Not bad! You did pretty well!");**

**}**

**else**

**{**

**Console.WriteLine("Better luck next time!");**

**}**

**Console.WriteLine("Thank you for playing Family Feud!");**

**Players receive feedback based on their total score, with different messages for various score ranges.**

**Conclusion**

**The FamilyFeudConsoleApp is a well-structured and interactive console game that mimics the format of the Family Feud TV show. The game provides a set of sample questions, handles user input gracefully, and calculates scores based on the player's answers. The code is clear and concise, making it easy to understand and modify for additional questions or features.**

**Possible Improvements**

**Randomizing Questions:**

**To enhance replayability, the questions could be presented in a random order.**

**Improving User Experience:**

**Providing hints or partial points for answers that are close to the correct ones.**

**Implementing a more sophisticated input validation to handle similar answers.**

**Extending Functionality:**

**Adding more questions and answers.**

**Introducing multiplayer mode for competitive play.**

**Overall, this console application is a fun and educational tool that showcases basic C# programming skills and provides an engaging user experience.**