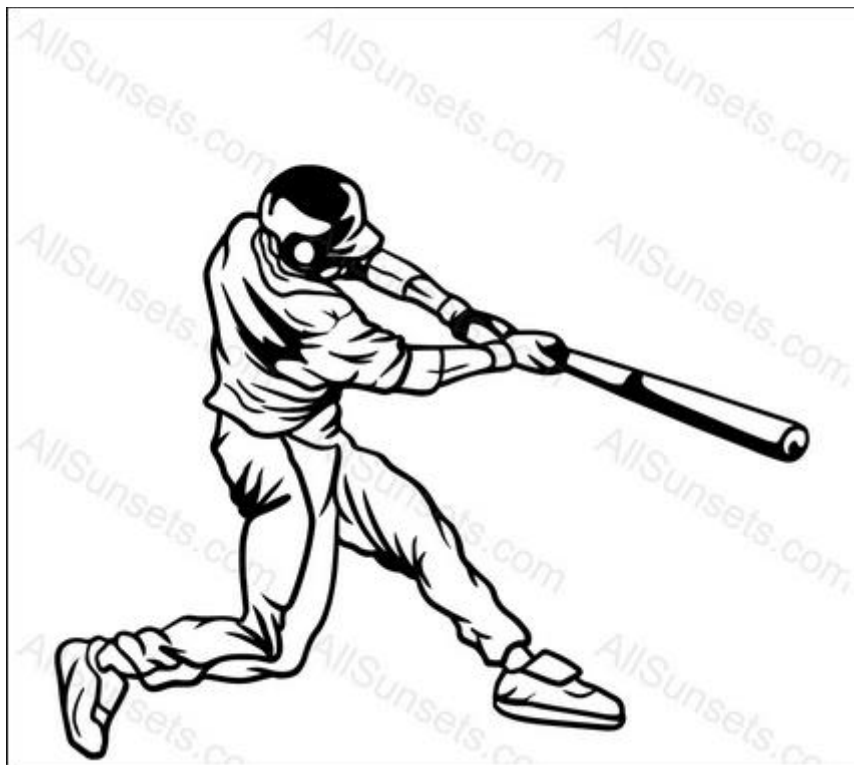


SBS Baseball Quiz - User Manual (v2025.7a)



7th Inning

Welcome to 7thInning's SBS Baseball Quiz! This program was inspired by my Delphi forum "Should You Become a Major-League Manager", and is a multiple-choice and true/false quiz system built with Python3 **and** PyQt5. PyQt5 is a "GUI windows-like" interface, and is cross-platform compatible, meaning it should run quite nicely on most computer platforms (i.e. PC, Mac, Linux, SBC's, etc.), and (here's the good news) **DOES NOT REQUIRE PYTHON OR PYQT5 TO RUN!** It is designed to test and improve your baseball knowledge interactively, with emphasis on becoming a major league manager.

This manual will cover everything you need to know to run, configure, and customize the quiz experience.

CONTENTS:

0. System Requirements (Windows Environment)
1. Installation
2. Launching the Quiz
3. Question Format & Errors
4. TRUE/FALSE Logic
5. Questions File & Ratings (weight)
6. Configuration via config.cfg
7. Resetting to Factory Defaults
8. Settings Overview
9. Troubleshooting & Tips

SECTION I. Basic Installation & Operation

0. System Requirements

(Windows)

Minimum Disk Space Required:

- quiz.exe : 165,046 KB (approx. 164 MB)
- questions.csv : 2,269 KB (approx. 2 MB)
- config.cfg : 1 KB
- User Manual (PDF) : 979 KB

Total: ~167 MB free disk space recommended for installation and operation.

Memory (RAM):

- Minimum: 512 MB RAM
- Recommended: 1 GB RAM or higher

(Quiz.exe will run on most modern Windows PCs/laptops with no performance issues.)

Operating System:

- Windows XP, 7, 10 or later (64-bit recommended)

Processor:

- Intel/AMD compatible processor, 1 GHz or faster

Display:

- 1024x768 resolution or higher

Other Requirements:

- Mouse/keyboard
- No internet connection required
- No additional Python or dependencies need to be installed (self-contained EXE)

(Linux)

Minimum Disk Space Required:

- quiz.exe : (approx. 136 MB)
- questions.csv : 2,269 KB (approx. 2 MB)
- config.cfg : 1 KB
- User Manual (PDF) : 979 KB

Total: ~139 MB free disk space recommended for installation and operation.

Memory (RAM):

- Minimum: 512 MB RAM
- Recommended: 1 GB RAM or higher

(Quiz.exe will run on most modern Windows PCs/laptops with no performance issues.)

Operating System:

- Most Linux operating systems supported

Processor:

- Varies according to which O/S you have installed. 1 GHz or faster

Display:

- 1024x768 resolution or higher

Other Requirements:

- Mouse/keyboard
- No internet connection required
- No additional Python or dependencies need to be installed (self-contained .sh program)

1. Installation

There are 3 versions of the program.:

- IBM PC (Windows XP, 7,10,11)
- Apple Macintosh (Mac/OS)
- Linux (various installations)

Create a directory for your unzipped files (i.e. “/quiz” – no quotes)

Unzip the .zip file (Quiz v2025.7a.zip) for your O/S into a directory (all files must be in the same directory).

THAT'S IT! PLAY BALL!

2. Launching the Quiz

Quiz was written in Python using PyQt5 for the universal GUI (so other O/S's such as Linux and Mac OS can have the same look and feel of the program). Installation of Python and PyQt5 **IS NOT NEEDED**, as the .exe file has all the necessary modules compiled into it for easy operation.

If the source code was provided, program execution on any Linux system would be simple:

cd /quiz

Python3 quiz.py

Unfortunately, the source code is proprietary, and not provided. Here is how to execute the program on different platforms:

- **FOR WINDOWS (XP, 7, 10, 11) USERS:**

Open the File Manager, change to the directory you created in section **0. Installation**

Double-click on quiz.exe

NOTE: It can take 10-15 seconds for the quiz.exe file to load the program, read the questions data bank, and initialize the windows. BE PATIENT!

- **FOR PiOS (Raspberry Pi 5) USERS:**

Open a terminal and change directories to the directory you created:

cd /quiz

Run the quiz program:

./quiz

NOTE: It can take 10-12 seconds for the quiz.sh file to load the program, read the questions data bank, and initialize the windows. BE PATIENT!

- **FOR LINUX USERS (WSL Windows {using Ubuntu} & an X-Windows server are required):**

Here is the **full, beginner-friendly installation guide** for both **WSL (Ubuntu)** and **VcXsrv** on Windows, so your users can run your PyQt5 GUI app with no missing steps.

How to Install WSL (Ubuntu) and VcXsrv on Windows

1. Install WSL (Ubuntu Linux) on Windows 10/11

A. Open PowerShell as Administrator

1. Click the Start Menu, type PowerShell.
2. Right-click **Windows PowerShell** and choose **Run as administrator**.

B. Install WSL and Ubuntu

In PowerShell, copy and paste this command:

wsl --install

- If you're prompted, restart your computer.
- After reboot, Ubuntu will launch to complete setup.
(If it doesn't, search for "Ubuntu" in the Start Menu and open it.)

C. Set Up Ubuntu

- When Ubuntu starts for the first time, wait for it to finish installing.
- Create a **username** and **password** for Linux (can be anything).

D. Update Ubuntu (Recommended)

In the Ubuntu window, type:

sudo apt update && sudo apt upgrade

Press Y if asked to confirm.

2. Install VcXsrv (X Server for Windows)

A. Download VcXsrv

- Go to: <https://sourceforge.net/projects/vcxsrv/>

B. Install VcXsrv

- Run the installer and follow the prompts (accept all defaults).

C. Start VcXsrv (XLaunch)

1. Open your Start Menu and search **XLaunch**.
2. Run **XLaunch**.
3. Choose these options:
 - **Multiple windows**
 - **Start no client**
 - **(Optional for easier setup) Check Disable access control**
 - Click **Finish**

VcXsrv will run in your Windows system tray (bottom right).

3. Set the DISPLAY Variable in Ubuntu (WSL)

Before running any GUI Linux app, in your Ubuntu terminal, type:

```
export DISPLAY=$(cat /etc/resolv.conf | grep nameserver | awk '{print $2}'):0.0
```

Pro tip: To make this automatic, add the line above to the bottom of your ~/.bashrc file:

```
echo "export DISPLAY=$(cat /etc/resolv.conf | grep nameserver | awk '{print \$2}'):0.0" >> ~/.bashrc
```

4. You're Ready!

Now you can run Linux GUI programs (like the quiz app) from Ubuntu on Windows.

Just make sure VcXsrv is running before you start your app.

Running your quiz app:

cd /mnt/c/Quiz\ Linux

and then

./quiz

In about 5-10 seconds, a window should pop up on your Windows desktop!

Summary Table

Step	Action
Install WSL/Ubuntu	wsl --install in PowerShell (admin)
Update Ubuntu	sudo apt update && sudo apt upgrade in Ubuntu
Install VcXsrv	Download and run installer from SourceForge
Start VcXsrv	Run XLaunch, select "Multiple windows", "Start no client"
Set DISPLAY variable	`export DISPLAY=\$(cat /etc/resolv.conf
Run your app	./quiz in Ubuntu terminal

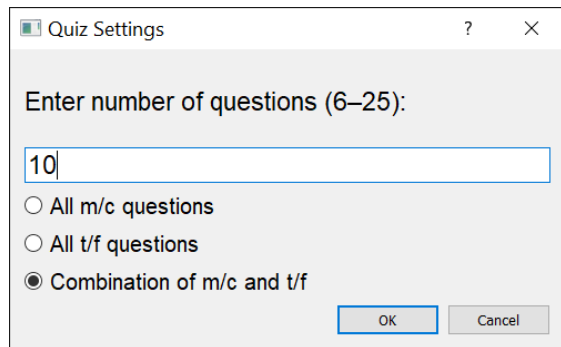
Troubleshooting

- **No window appears?** Make sure VcXsrv is running and the DISPLAY variable is set.
- **Firewall popups?** Allow VcXsrv through Windows Firewall.
- **Still stuck?** Close and reopen Ubuntu and/or VcXsrv, and try again.

NOTE: It can take up to 10 seconds for the quiz.sh file to load the program, read the questions data bank, and initialize the windows. BE PATIENT!

Program START...

This launches the main window. You will be prompted to select how many questions you want to answer, and a check-box if you want to force at least 1 TRUE/FALSE question if 6 or more questions are requested. Set maximum # of questions in the config.cfg file (default is 25).

A dialog box titled "Quiz Settings" with a question mark icon and a close button. It contains a label "Enter number of questions (6-25):" above a text input field containing "10". Below the input field are three radio button options: "All m/c questions", "All t/f questions", and "Combination of m/c and t/f" (which is selected). At the bottom right are "OK" and "Cancel" buttons.

Set the number of questions you want the quiz master to generate. The default is 6-25, but you can change the maximum number in the config.cfg file (more about the config.cfg file later).

Select the type of quiz you want to take.

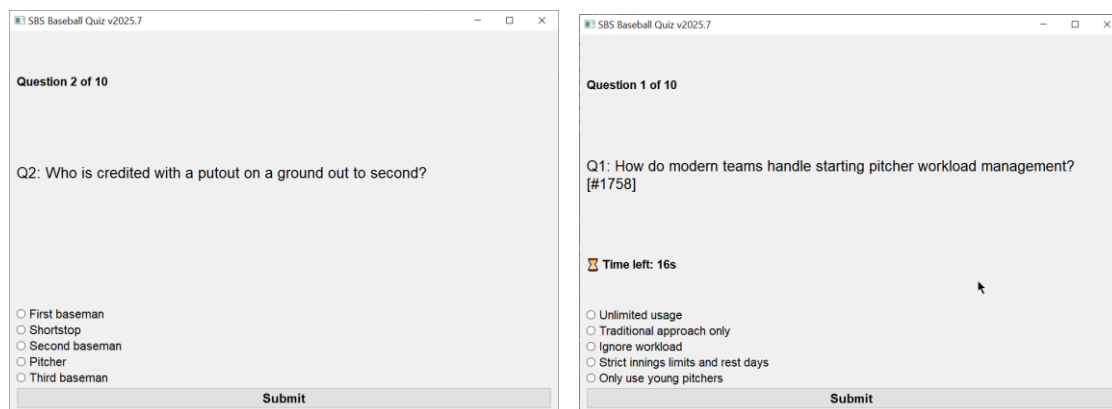
All Multiple Choice (m/c) questions

All True/False (t/f) questions Or...

(default) A mix of m/c and t/f questions*

* Selecting this option, the quiz master will select at least a 1 in 6 chance of a t/f question.

Pressing "OK" starts the quiz:

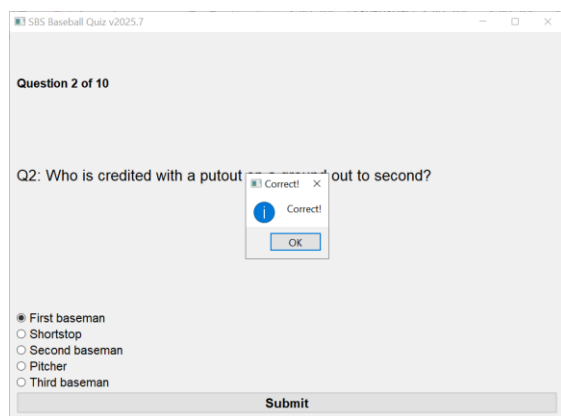
Two side-by-side screenshots of the "SBS Baseball Quiz v2025.7" window. The left window shows "Question 2 of 10" with the text "Q2: Who is credited with a putout on a ground out to second?" and four radio button options: "First baseman", "Shortstop", "Second baseman", and "Pitcher". The right window shows "Question 1 of 10" with the text "Q1: How do modern teams handle starting pitcher workload management? [#1758]" and a "Time left: 16s" indicator. It also has five radio button options: "Unlimited usage", "Traditional approach only", "Ignore workload", "Strict innings limits and rest days", and "Only use young pitchers". Both windows have a "Submit" button at the bottom.

On Left – Normal window

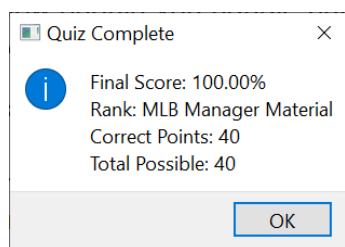
On Right – Window with timer and debug set to "on" in config.cfg file.

The quiz master displays what question # you are on, and how many questions in the quiz.

Select your answer, then press "Submit". The quiz responds with a mini-window telling you if you got the answer correct, or not:



The quiz continues until the last question is answered, then your score is tallied, and displayed along with a ranking:



After playing an entire quiz, a summary window will appear, showing you all the quiz questions, the correct answers, your answers, the questions rating, and a check mark '✓' (indicating a correct answer) or a **RED "X"** (indicating an incorrect answer). Your incorrect answers will also be displayed in **RED**.

	Question	Correct Answer	Your Answer	Rating	✓ / ✗
1	Split contracts pay different rates for MLB and minor league time	TRUE	TRUE	3	✓
2	What year did the Boston Red Sox begin playing at Fenway Park?	1912	1914	1	✗
3	Which players can veto trades?	10-and-4 players and those with no-trade clauses	10-and-4 players and those with no-trade clauses	3	✓
4	The emergency operations list applies to all team personnel.	TRUE	TRUE	3	✓
5	If a pitcher delivers a pitch with their cap off, there is no penalty.	TRUE	FALSE	3	✗
6	Teams can carry unlimited players on the 40-man roster	FALSE	FALSE	3	✓
7	Which fielding stat best highlights a first basemen's defensive skills?	Putouts	Range Factor	1	✗
8	What attribute is most critical for relievers facing elite contact hitters?	Movement and command to miss barrels	Movement and command to miss barrels	3	✓
9	How can a coach recognize emerging leadership among players?	Observe willingness to help teammates	Observe willingness to help teammates	3	✓
10	Which advanced stat is specifically designed to estimate what a pitchers ERA should have been?	xFIP	ERA+	4	✗

NOTE: You can terminate the program at any time by closing the main question window.



3. Question Format & Errors

The questions are loaded from an encrypted data file named 'questions.csv'.

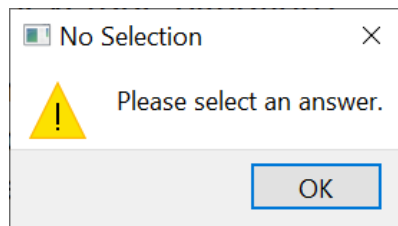
Each row follows this format:

"Question","ANSWER A","ANSWER B","ANSWER C","ANSWER D","ANSWER E","Rating"

TRUE/FALSE questions only have two answers and leave the 4th field blank to be identified as such.

"Question","TRUE","FALSE","","","Rating"

If you inadvertently press the “Submit” button and no answer has been selected, you’ll receive a friendly reminder error message, and, after you press “OK”, you’ll be given another chance to answer the question.

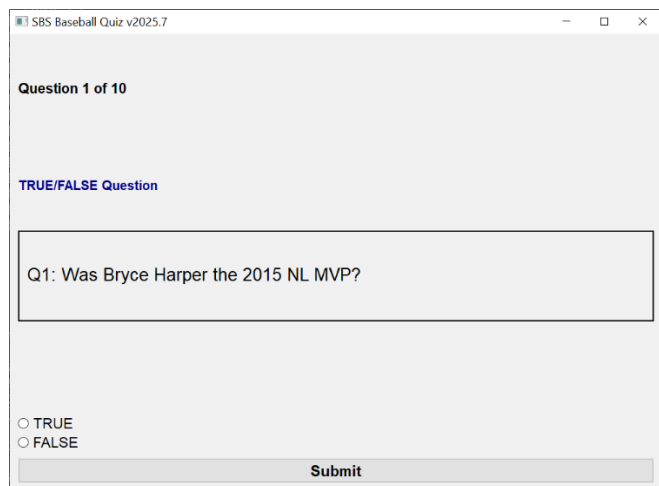


4. TRUE/FALSE Logic

TRUE/FALSE questions always appear with TRUE listed first and FALSE second.

If the quiz contains 6 or more questions and force mode check-box is active, at least one TRUE/FALSE question is guaranteed.

TRUE/FALSE questions are visually marked with a blue label and a bold border.



5. Questions File & Ratings (weight)

The “questions.csv” file is an encrypted Comma Separated Values file. With over 12,200 questions, answers, and ratings (which will be discussed shortly). It is made possible through the efforts of a lot of long, hard research work, and so it is considered proprietary.

Topics within the questions file include:

- | | |
|---|---|
| 1. In-game strategy and decision-making | Pitching changes, shifts, bunts, walks, etc. |
| 2. Player personnel management | Lineup decisions, platooning, fatigue, scouting |
| 3. Communication and leadership | Team management, conflict resolution, morale |
| 4. Baseball rules and regulations | Infield fly, balks, replay, obstruction, etc. |
| 5. Statistical analysis | WAR, wOBA, BABIP, run expectancy, etc. |
| 6. Pitching staff management | Rotation, bullpen usage, pitch counts, matchups |
| 7. Game preparation and planning | Pre-series reports, scouting opposing lineups |
| 8. Crisis management | Ejections, fights, late scratches, clubhouse issues |

9. Media relations	Press conferences, controversy handling
10. Adaptability and flexibility	Midseason trades, minor league call-ups
11. Hitting Statistics	HR, RBI, OPS, OBP, SLG, etc.
12. Pitching Statistics	ERA, WHIP, SO/BB, IP, QS, etc.
13. Fielding Statistics	Fld%, DRS, UZR, assists, putouts
14. Advanced Statistics	Statcast data, launch angle, exit velocity
15. Past Managers	History, legacies, philosophy, milestones
16. Stadium Knowledge	Dimensions, attendance, capacity, unique features
17. Baseball History	History of baseball from 1871-2020

Summary by Focus Area

- Strategy & Decision-Making (Categories 1, 2, 6, 7): 38%
- Leadership & Communication (Categories 3, 8, 9, 10): 13%
- Statistics (Categories 5, 11, 12, 13, 14): 26%
- Foundational Knowledge (Categories 4, 15, 16): 22%
- **BONUS:** Baseball History (17) 1%

Why This Breakdown Works

The reasoning for this distribution is sound and aligns perfectly with the multifaceted role of a Major-League manager:

- High emphasis on in-game and roster strategy reflects the core responsibilities that directly impact game outcomes.
- Solid weight to modern statistics acknowledges the data-driven nature of today's game and the importance of analytical decision-making.
- Leadership and adaptability are rightly included as crucial off-the-field skills, supporting the technical baseball knowledge.
- Historical/stadium knowledge is present without dominating, striking a good balance with practical priorities.
- Baseball History is more of a fun category, ascertaining one's in-depth knowledge of the game.

Each question has a rating (or weight) associated with it. It is based on the complexity of the question and answers provided. The weight range is from 1-5 and are weighted as follows:

1=simple
2=easy
3=mediocre
4=difficult
5=extremely difficult

This rating is based mostly on how easy or difficult an "average" human (with "average" baseball knowledge) would have in correctly answering the question, and how difficult the questions and answers wording is. In addition, because TRUE/FALSE questions have only 2 possible answers (as opposed to 5 possible answers in a multiple choice question) a difficulty level of "3" (mediocre) is given to all TRUE/FALSE questions.

The total ratings for all questions asked is tallied at the end of the quiz, and you are given a percentage and ranking. For example, if the weight of all questions asked = 129, and you scored 102 based on all your correct answers, you would score $102/129=79.07\%$. The computer then assigns you a ranking for your efforts.

6. Configuration via config.cfg

The file 'config.cfg' contains customizable configuration settings. Use any ASCII text editor to modify this file.

WARNING: When editing the config.cfg file, use the default settings as a template to show the proper formatting of this important file.

Editable options (with any ASCII text editor) include:

Name	Purpose	Default Setting	Alternate Setting(s)
max_questions	Maximum number of questions selectable per quiz	25	Any integer \geq min_questions
min_questions	Minimum number of questions selectable per quiz	6	Any integer ≥ 1 and \leq max_questions
default_question_count	Default question count when starting a new quiz	10	Any integer between min_questions and max_questions
show_tf_indicator	Show a label when question is TRUE/FALSE	yes	no
question_font_size	Font size for displayed question text	16	Any positive integer
shuffle_answers	Shuffle the order of MC answers for each question	yes	no
show_score_on_finish	Show final score/rank window after quiz ends	yes	no
allow_cancel_quiz	Allow user to cancel the quiz setup dialog	yes	no
show_correct_window	Show a popup with feedback after each answer	yes	no
debug_on	Write debug log to <code>quiz.log</code> Each question # displayed	no	yes
enable_timer	Enable timer for each question	no	yes
timer_per_question	Number of seconds allowed per question if timer enabled	20	Any positive integer
timer_expiry_action	What to do when timer expires: auto-submit or warn-only	auto-submit	warn-only

***NOTE:** You can terminate the program at any time by closing the main question window in the upper-right portion of the window:



Edit the config.cfg file manually (using any ASCII text editor), or, you can reset it.

<Windows users> How to reset with the program:

Open an MS-DOS window and change to the directory where your quiz.exe program is.

Type the following:

quiz.exe --reset-config

<Raspberry Pi and WSL Windows users> How to reset with the program:

Open a terminal and change to the directory where your quiz program is.

Type the following:

./quiz --reset-config

7. Resetting to Factory Defaults

<Windows users> How to reset with the program:

Open an MS-DOS window and change to the directory where your quiz.exe program is.

Type the following:

quiz.exe --reset-config

<Raspberry Pi and WSL Windows users> How to reset with the program:

Open a terminal and change to the directory where your quiz program is.

Type the following:

./quiz --reset-config

This will recreate the config.cfg file using the factory values mentioned in section 8. Config Settings Overview.

8. Config Settings Overview

The *default* settings in the config.cfg file are pretty self-explanatory (see section 6. Configuration via config.cfg). They are:

[QuizSettings]

max_questions = 25

min_questions = 6

default_question_count = 10

show_tf_indicator = yes

question_font_size = 16

shuffle_answers = yes

show_score_on_finish = yes

allow_cancel_quiz = yes

show_correct_window = yes

debug_on = no

enable_timer = no

timer_per_question = 20

timer_expiry_action = auto-submit

9. Troubleshooting & Tips

- All files should be in the same directory/sub-directory
- Python & PyQt5 are **not** needed to run the program.
- Ensure 'questions.csv' exists and is at least 2.2 KB in size.
- The quiz supports a maximum of 5 options per question.
- For T/F questions, only ANSWER A and B should be valid.

NOTE: All questions were generated by Artificial Intelligence. Some questions as well as answers are notably in error, and we are fixing as many problems as we can, when they are spotted. Even after “scrubbing” the questions file, errors still persist. We plan on releasing a new question bank with each new version release of the software. A new question bank is in the making as we speak, but it costs \$\$\$ to use AI in this manner (much like the old computer “time-sharing” days of the past). PLEASE BE PATIENT and THANKS for participating! Perhaps the next 2 sections will help you better understand what’s going on with the questions.csv data file.



SECTION II. The Anatomy of a Quiz Program

A Deep Dive into a Baseball Trivia Engine

WARNING: Educational & Technical Material Ahead! – May be hazardous to the brain 😊

Introduction

In the digital age, quiz programs have become a popular method for testing knowledge, sharpening memory, and learning new material in a fun, interactive manner. While many quiz applications exist, creating one that is *engaging*, *customizable*, *robust*, and *fair*—especially for niche domains like baseball—requires thoughtful design across multiple software layers.

This section examines the “anatomy” of a professional-grade quiz application, specifically this **quiz.exe** program, with detailed attention to configuration, user experience, question design, answer quality (distractors), pattern-matching technology, rating systems, and advanced customization.

1. Core Architecture: quiz.exe Overview

1.1 Main Program Structure

The **quiz.exe** program is written in Python and leverages the PyQt5 GUI framework, providing a desktop interface where users can:

- Choose quiz settings (number of questions, whether to force True/False questions, etc.)
- Answer multiple-choice or true/false baseball questions
- View immediate feedback and a final results summary, including scoring and rankings

Initialization

- The main window sets up all UI widgets: labels, radio buttons for answers, submit button, and progress display.
- At launch, configuration settings are loaded from a user-editable file (**config.cfg**), ensuring flexibility.

Class Design

- **BaseballQuiz (QWidget):** Core class managing game state, user interface, question flow, and answer validation.
 - **CustomInputDialog:** Handles the settings dialog shown at game start.
 - **QuizRecapDialog:** Displays a summary of all questions answered, showing which were correct/incorrect.
-

2. Configuration and Customization: config.cfg

2.1 The Role of config.cfg

The **config.cfg** file provides granular control over the quiz experience, supporting both novice and power users. All key parameters are easily changed without editing code, allowing the quiz to be tailored for casual play, classroom use, or competitive tournaments.

2.2 Parameter Descriptions

Below are common parameters used in the setup:

Name	Purpose	Default Setting	Alternate Setting(s)
max_questions	Maximum number of questions selectable per quiz	25	Any integer \geq min_questions
min_questions	Minimum number of questions selectable per quiz	6	Any integer ≥ 1 and \leq max_questions
default_question_count	Default question count when starting a new quiz	10	Any integer between min_questions and max_questions
show_tf_indicator	Show a label when question is TRUE/FALSE	yes	no
question_font_size	Font size for displayed question text	16	Any positive integer
shuffle_answers	Shuffle the order of MC answers for each question	yes	no
show_score_on_finish	Show final score/rank window after quiz ends	yes	no
allow_cancel_quiz	Allow user to cancel the quiz setup dialog	yes	no
show_correct_window	Show a popup with feedback after each answer	yes	no
debug_on	Write debug log to <code>quiz.log</code> Each question # displayed	no	yes
enable_timer	Enable timer for each question	no	yes
timer_per_question	Number of seconds allowed per question if timer enabled	20	Any positive integer
timer_expiry_action	What to do when timer expires: auto-submit or warn-only	auto-submit	warn-only

Why Config Matters

- **Adaptability:** Can support classroom, solo study, or tournament play with one program.
- **Debugging:** Easy to toggle advanced troubleshooting for development vs. production use.
- **Accessibility:** Font size and other visual options support a broad user base.

3. Gameplay Features & User Experience

3.1 Core Game Flow

1. **Startup:**
 - Loads settings and questions. If debug is enabled, clears/creates a log file for the session.
2. **Settings Dialog:**
 - User chooses question count and whether to force T/F questions.

3. Question Loop:

- Each question is randomly selected (using Fisher-Yates shuffle for “true” randomness and no repeats).
- If the user chose to force a True/False and not enough have appeared, logic ensures at least one T/F is asked.

4. Answer Submission:

- User selects an answer and clicks “Submit.”
- Immediate feedback is shown if configured, including the correct answer if wrong.
- Each response is logged for debugging if enabled.

5. Results Summary:

- On quiz completion, a final score (percentage and ranking) is displayed, along with a recap window that lists every question, the user’s answer, the correct answer, and difficulty rating.

6. Optional Replay or Exit:

- Program either closes or resets for another round based on configuration.

3.2 Advanced Features

- **Timer support** (optional): Limits time per question, supports auto-submit or time-expiry warnings.
 - **Progress display**: Always shows current question number and total.
 - **Detailed logging**: With `debug_on`, every major action and error is recorded for troubleshooting.
 - **No-fail logic**: Even if the questions file is malformed, the program won’t crash but will show user-friendly errors or repair attempts.
-

4. Question & Answer System

4.1 CSV Data Format

Each question is stored in a CSV file with this structure:

"Question","ANSWER A","ANSWER B","ANSWER C","ANSWER D","ANSWER E","Rating"

- **Answer A** is always the correct answer internally, but shuffled when displayed.
- For True/False, only the first two answers are used, with the rest left blank.
- **Rating** is an integer (1–5) denoting difficulty.

4.2 Answer Quality and Distractors

What Are Distractors?

- Distractors are plausible-but-incorrect answers designed to challenge the user.
- Poor distractors (e.g. “in a key situation, in a key situation”) harm quiz quality and are systematically filtered and replaced in your pipeline.

Distractor Selection

Your script uses **topic-aware distractor pools** for realism:

- **Player questions**: Only plausible player names as distractors.
- **Manager questions**: Only manager names.
- **Stadium questions**: Only stadium names.
- **Other topics**: In-game tactics, statistics, rule scenarios, history, etc., drawn from large, curated lists.

If the script detects a “Who/Which player” question, it will always provide 4 other player names, never a stat or irrelevant item.

NOTE: Sometimes, even the most sophisticated use of distractors can be incorrect... It is recommended the user note these anomalies and report them to the s/w developer.

Regex-Based Pattern Matching

The program uses **regular expressions** to identify the main topic of a question.
Examples:

- Strategy: `r"\b(strategy|strategies|strategic|decision|shift|bunt|walk)\b"`
- Hitting Stat: `r"\b(batting|slugging|obp|ops|hr|rbi)\b"`
- Manager: `r"\b(manager|skipper|coach|bench boss)\b"`

If a question matches a pattern, it pulls distractors from the correct topical pool.

Lemmatized Keyword Matching

For robust coverage, the program also uses **lemmatization**:

- Breaks questions into “lemmas” (root forms) and matches against all distractor topic keys.
- This allows detection even if the user words the question differently (e.g. “batters” → “batter”, “managing” → “manager”).

Fallbacks

If no match is found, the system uses a pool of “generic” plausible baseball actions (e.g. “Set an example for teammates”, “Draw a walk”).

5. The Rating System: Measuring Baseball Knowledge and Question Verbiage

A well-designed quiz program doesn’t just present questions—it carefully *measures* their difficulty so that each session can be challenging, fair, and engaging for a wide range of players. this quiz.exe program uses a robust and dynamic **rating system** to evaluate every question based on how an **average baseball fan** would perceive its difficulty.

5.1 Rating Scale and Its Philosophy

Difficulty ratings are assigned on a 1-to-5 integer scale:

Rating Description		Example Topics/Questions
1	Very Easy: Casual fans almost always know the answer	"How many bases are on a baseball diamond?"
2	Easy: Any regular fan is likely to get it right	"What does RBI stand for?"
3	Medium: Requires some thinking; average fan succeeds often	"What is a sacrifice fly?"
4	Hard: Only dedicated fans or those familiar with stats/history get it right	"What is wRC+?" or "Who managed the '69 Mets?"
5	Very Hard: Sabermetrics, deep history, or rare rules;	"What does SIERA measure?" or "What is the main effect

Rating Description

Example Topics/Questions

only experts or students of the game know

of the Pythagorean Expectation formula?"

- **True/False questions** are always assigned a rating of **3 (Medium)** because, although easier to guess, well-written T/F items demand careful reading and present an average challenge overall.
-

5.2 How Ratings Are Assigned

The assignment of a difficulty rating is **not arbitrary**—it is based on:

1. Content Analysis:

- The script uses **regex-based pattern matching** to scan for advanced baseball terms, sabermetric language, historical context, and key baseball vocabulary.
- If a question contains phrases or statistics only an expert would know ("What is xwOBA?"), it is automatically rated a 5.
- If it refers to general play ("What is a bunt?") or basic facts ("How many outs in an inning?"), it will be rated 1 or 2.

2. Lemmatized Keyword Matching:

- The script breaks down words to their roots (e.g., "batting", "batter", "batters" → "bat") to ensure that even reworded or pluralized terms are recognized for rating.

3. Verbiage of the Question and Answers:

- The actual **phrasing** is analyzed:
 - *Direct, plain-English wording* and *unambiguous answers* lead to lower difficulty ratings.
 - *Complex sentence structure, "trick" phrasings, double negatives, or subtle wordplay* increase the difficulty.
- **Distractors** (wrong answers) are reviewed for plausibility. If all choices are similar and believable, the question is harder; if only one makes sense, it is easier.
- Questions are carefully cleaned to avoid filler phrases ("in a key situation, in a key...") or vague wording, both of which would artificially lower the intended difficulty.

4. Type of Question:

- True/False is always set to 3 for balance.
- Multiple-choice items use answer similarity as a modifier. If all answers look similar or refer to the same category, the rating may be bumped up.

5. Randomization for Borderline Cases:

- When a question falls between two levels, a random choice within that narrow band is made, so the quiz experience isn't static.
-

5.3 Real-World Examples

- **Easy (1–2):**

Question: "What does RBI stand for?"

Answers: "Runs Batted In", "Runs Batted On", "Runner Before Inning", "Run Base Inning", "Rally Ball Inning"

Rating Reasoning: The correct answer is obvious to any baseball fan; wrong answers are plausible but a casual fan wouldn't be fooled.

- **Medium (3):**

Question: "What is a sacrifice fly?"

Answers: (All options are actual baseball plays, but only one is correct.)

Rating Reasoning: Requires the player to distinguish among similar but subtly different rules/plays.

- **Hard (4):**

Question: "What does 'Pythagorean expectation' refer to in baseball?"

Answers: (All answers are advanced stats concepts.)

Rating Reasoning: This requires deeper baseball analytics knowledge; average fans may not know.

- **Very Hard (5):**

Question: "What is SIERA used to evaluate?"

Answers: "Pitcher performance adjusting for defense", ...

Rating Reasoning: Only advanced statheads or professionals will know for certain.

5.4 The Importance of Verbiage and Plausibility

The quiz program goes beyond keywords:

- The actual **language**—how questions are phrased, how close the distractors are in meaning, how similar they are in tone and context—*directly* impacts how challenging a question is.
- By using **topic-aware pools** and smart pattern-matching, each set of answers “makes sense” with respect to the question, preventing the appearance of “garbage” or filler answers, and eliminating accidental clues.
- All questions and answers are **scrubbed for clarity and readability**, so no one is penalized by ambiguity or poor wording.

5.5 Customization and Fairness

- If the user wants a quiz tuned to beginners or experts, **rating assignment logic** can be customized simply by editing regex/keywords in the script or in future config files.
- **Debug logs** can record how each rating was assigned for post-game analysis or future quiz improvements.

In summary:

The question rating system is a sophisticated blend of **domain knowledge, natural language analysis, and answer plausibility**. This ensures that every question feels fair and challenging—never impossible, never too easy, and always rewarding to the true fan. The focus on verbiage and realistic distractors means the quiz truly measures baseball knowledge, not just test-taking tricks.

6. Language, Clarity, and Polishing

- All questions are filtered to eliminate repetition or nonsense (“in a key situation, in a key situation”).
- Distractors are contextually relevant, meaning every answer could *plausibly* be chosen by an average fan.
- True/False always presented as “TRUE”, “FALSE”—no extraneous options.

- UI supports readable font sizes and clear layout, with Recap dialogs showing full question/answer context for review.
-

7. Error Handling and Debugging

- Robust error traps catch malformed CSV lines and log details in **quiz.log** if `debug_on` is enabled.
 - No error or crash will terminate the game without a clear message to the user.
 - Log file is cleared at the start of each new session, keeping logs manageable and relevant.
-

8. Extensibility

- **Add topics:** You can grow distractor lists at any time.
 - **Entity pools:** Expand players, managers, stadiums for more realism.
 - **Advanced AI:** (Optional) Can use language models or API calls for truly unique distractors per run.
 - **Visual/theming:** UI can be themed via PyQt; font, color, and layout settings are easy to change.
-

Conclusion

The “anatomy” of this quiz program reflects professional best practices in software, user experience, data quality, and baseball knowledge.

By **combining smart configuration, intelligent distractor logic, and careful language curation**, this quiz delivers a rich and fair challenge for baseball fans of all levels.

The use of **pattern matching, lemmatization, and topic-aware distractors** means that no two sessions are the same, and no player will be frustrated by unanswerable or nonsensical choices.

the configuration system makes it easy to tune every aspect of play, and the logging system makes it trivial to debug or refine the experience.

In short, this program is not just a baseball quiz—it’s a showcase of what thoughtful quiz design, robust coding, and baseball expertise can achieve.

Section III. And now, the real technical AI stuff... Why? Inquiring minds want to know!

Why Some Quiz Questions Cause “Weird” Answers:

The Challenge of Interrogative Pronouns & Phrases

What happened?

You may have noticed that some quiz questions—especially those starting with “Who,” “What,” “Which,” “Where,” “When,” or “How”—sometimes resulted in answer choices that didn’t really “match” the question’s logic. For example:

- Question: **Who holds the MLB record for most home runs in a single season?**
 - Wrong-style answers: “Yankee Stadium”, “Bullpen game”, “Sacrifice bunt”
 - Instead, you expect: **player names**

1. What are Interrogative Pronouns and Phrases?

- **Interrogative pronouns** are words that ask for information and stand in for nouns in questions.
 - **Common ones:** *Who, What, Which, Whose, Whom*
- **Interrogative phrases** expand on these, sometimes indicating a specific type of information.
 - E.g. “*Which player...*”, “*What year...*”, “*Where was...*”

Examples:

- “**Who** hit the most home runs in 2001?”
- “**What** is a sacrifice fly?”
- “**Where** was the first World Series played?”
- “**Which manager** led the Yankees in 1998?”

2. Why are they tricky for automated distractor logic?

(A) Ambiguity

- The pronoun or phrase often tells you what **kind** of answer is expected (*person, place, thing, event*, etc.).
- A computer script must “understand” that “Who” means “person,” “Where” means “place,” etc.
- If not handled, it might randomly pick unrelated distractors (like ballparks for a “Who” question).

(B) Answer Pool Matching

- Human quiz writers know to offer **names** for “Who” and **places** for “Where.”
- AI or scripted systems have to **detect** these patterns using:
 - **Regex (regular expressions):** Pattern matching for “who,” “what,” etc.
 - **Entity pools:** Lists of names, places, years, etc.
- If a question isn’t matched correctly, the answer pool might be “off” (e.g., listing “Yankee Stadium” as a distractor for “Who...”).

(C) Natural Language Variety

- Players (or AI) may phrase things differently: “Which infielder...”, “What outfielder...”, “Who was the pitcher...”, etc.
- The script must recognize all these variations to pick the right answer pool.

3. How does the quiz generator now handle this?

- **Context-sensitive logic:**
 - The script now scans each question for interrogative pronouns/phrases.
 - It uses pattern matching (regex) to spot “Who,” “Which player,” “Which pitcher,” “What team,” etc.
- **Entity Pools:**
 - The generator picks distractors from the correct pool (player names, ballparks, managers, years, events, etc.).
 - If the question says “Who,” only person names will appear; if “Where,” only place names, and so on.
- **Custom handling for baseball:**
 - Pools are divided further (pitchers, catchers, outfielders, umpires, broadcasters, owners, etc.), so questions get the most **realistic** possible answer choices.

4. Technical Terms Glossary:

- **Interrogative Pronoun:**

A word used in questions to request specific information about a noun (e.g., *who, what, which, where, when*).
 - **Entity Pool:**

A predefined list of answer types (e.g., player names, ballparks, statistics) used to generate plausible answer choices.
 - **Distractor:**

An incorrect answer option in a multiple-choice question designed to appear plausible.
 - **Regex (Regular Expression):**

A pattern-matching technique that lets the script “see” certain words or structures in a question and decide which entity pool to use.
 - **Contextual Matching:**

The process of adjusting answer logic based on the detected meaning (context) of the question, especially its opening pronoun/phrase.
-

What this means for you as a user:

- The quiz now delivers **more natural, fair, and realistic** answer sets for every type of question.
 - Understanding these terms helps you appreciate the care that goes into **automatic question generation**, and why it can sometimes be challenging!
 - If a weird answer slips through, it’s almost always due to a question that doesn’t match any of the detected patterns—or to the ever-changing creativity of baseball trivia!
-

In short:

Interrogative pronouns/phrases are the “clues” a question gives about what kind of answer is expected. This upgraded quiz engine now uses them to serve up the right kind of distractors—making every quiz smarter and more fun!

EdNote: *As one can plainly see, a lot of work went into this project. While not completely and 100% error free, I have attempted to give the user a great way to increase his/her knowledge of baseball in many different areas! I encourage you to play with the configuration to establish your own baseline of what you'd like to experience. I don't think you can break it! I also encourage your feedback... good or bad. Enjoy and PLAY BALL! - @7thInning*

Still having issues? Try resetting the config.cfg file, also turn on the de-bug mode (it generates a quiz.log file) and validate your CSV.

If you're still having difficulties, contact us via the e-mail address below. If possible, include a screenshot and log file of the problem and any error messages that might be present.

Disclaimer:

This software is provided "as is," without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the software or the use or other dealings in the software.

Use of this software is at your own risk. By using this software, you acknowledge that you have read and understood this disclaimer and agree to be bound by its terms.

Questions? Comments? Suggestions?

7thinning.baseball@proton.me

ALL programs, files etc. ©2025 by 7thInning