

🌟 Week 2 Student Guide – Advanced Stats Calculator

This guide provides a structured walkthrough for your Week 2 assignment, focusing on **Control Flow** (if/else statements and loops) and **Functions** to calculate advanced basketball efficiency metrics.

🏀 Part 1: Reviewing Core Concepts

This week's objective is to apply Python's control flow and functions to calculate two key player efficiency stats: **Effective Field Goal Percentage (eFG%)** and **True Shooting Percentage (TS%)**.

1. Functions for Advanced Metrics

Functions allow you to package a piece of code that can be reused with different inputs (parameters). They are essential for calculating complex, multi-step metrics.

Metric	Purpose	Formula (in Python)
Effective FG% (eFG%)	Adjusts Field Goal Percentage (FG%) to account for the value of a three-pointer, giving it a \$1.5 \times\$ value.	$\text{eFG\%} = \frac{\text{FGM} + 0.5 \times \text{3PM}}{\text{FGA}}$
True Shooting % (TS%)	The most complete measure of scoring efficiency, accounting for field goals, three-pointers, and free throws.	$\text{TS\%} = \frac{\text{PTS}}{2 \times (\text{FGA} + 0.44 \times \text{FTA})}$

Note on TS% Formula: The $2 \times (\text{FGA} + 0.44 \times \text{FTA})$ denominator is an estimate of a player's **True Shooting Attempts (TSA)**, or the number of scoring possessions they used. The **\$0.44\$** factor is an approximation to account for the fact that not all free throw attempts end a possession (e.g., "and-one" free throws or technical free throws).

2. Control Flow: If/Else and Loops

- **If/Else Statements (Conditionals):** Used to execute different blocks of code based on whether a condition is True or False. This is perfect for classifying a player's efficiency.

Python

```
if efficiency > 0.6:
    print("Elite")
elif efficiency >= 0.5:
    print("Above Average")
```

```
else:  
    print("Below Average")
```

- **For Loops:** Used to iterate over a sequence (like a list of players) and run the same code for each item. This is the key to solving the **Extra Challenge**.



Part 2: Week 2 Assignment Walkthrough

Follow the steps below to complete the assignment notebook.

Step 1: Define Functions (Code Cell 3)

You must define your two core functions. The function should accept the necessary box score stats as parameters and return the calculated percentage.

Required Formulas:

- **Effective FG%:** $(FGM + 0.5 * threePM) / FGA$
- **True Shooting %:** $PTS / (2 * (FGA + 0.44 * FTA))$

Python

```
# Define a function to calculate eFG%  
def effective_fg(FGM, threePM, FGA):  
    # Your code here  
    return (FGM + 0.5 * threePM) / FGA  
  
# Define a function to calculate TS%  
def true_shooting(PTS, FGA, FTA):  
    # Your code here  
    # Adding a check for division by zero to be safe  
    if FGA + 0.44 * FTA == 0:  
        return 0.0  
    return PTS / (2 * (FGA + 0.44 * FTA))
```

Step 2: Input Stats and Calculate (Code Cell 4)

Use the example stats provided in the notebook and call the functions you just defined. Remember to store the results in variables.

Python

```
# Example stats: Player line  
FGM = 10  
FGA = 20  
threePM = 3  
FTM = 6  
FTA = 8  
PTS = 29  
  
# Call your functions here and print the results  
player_efg = effective_fg(FGM, threePM, FGA)  
player_ts = true_shooting(PTS, FGA, FTA)
```

```
print("Effective FG%:", round(player_efg, 3))
print("True Shooting %:", round(player_ts, 3))
```

Expected Output (rounded to 3 decimal places):

- Effective FG%: **0.575**
- True Shooting %: **0.718**

Step 3: Efficiency Message (Code Cell 5)

Replace the placeholder value for efg with your actual calculated variable and use your **if/else statement** to print a customized message.

Python

```
player_efg = effective_fg(FGM, threePM, FGA) # Using the result from Step 2
```

```
if player_efg > 0.6:
    print("Elite shooting night! (eFG% > 0.60)")
elif player_efg >= 0.5:
    print("Above average efficiency. (eFG% >= 0.50)")
else:
    print("Below average efficiency. (eFG% < 0.50)")
```

Expected Output for the example stats:

- player_efg is 0.575, so the output should be: **"Above average efficiency. (eFG% >= 0.50)"**



Part 3: Extra Challenge – Multiple Players

This step requires a **loop** to process a list of player data and a logic to track and print the player with the highest efficiency.

Extra Challenge Solution (Code Cell 6)

You will iterate over the players list, calculate the TS% for each player, store the result back into the player's dictionary, and keep track of the most efficient player found so far.

Python

```
# Example: List of players with box score lines
```

```
players = [
    {"name": "Player A", "FGM": 10, "FGA": 20, "3PM": 3, "FTM": 6, "FTA": 8, "PTS": 29},
    {"name": "Player B", "FGM": 5, "FGA": 12, "3PM": 1, "FTM": 2, "FTA": 2, "PTS": 13},
    {"name": "Player C", "FGM": 8, "FGA": 15, "3PM": 2, "FTM": 4, "FTA": 5, "PTS": 22},
]
```

```
most_efficient_player = None
```

```
highest_ts = -1 # Initialize with a value lower than any possible TS%
```

```
# Loop through players, calculate stats, and print results
for player in players:
    # Calculate TS% using the defined function
    ts_percent = true_shooting(player["PTS"], player["FGA"], player["FTA"])

    # Store the calculated stat back in the dictionary (optional, but useful!)
    player["TS%"] = round(ts_percent, 3)

    print(f'{player["name"]}\'s TS%: {player["TS%"]}')

# Check for the most efficient player
if ts_percent > highest_ts:
    highest_ts = ts_percent
    most_efficient_player = player["name"]

print("\n---")
print(f"The most efficient player (by TS%) is: **{most_efficient_player}** (TS%: {round(highest_ts, 3)})")
```