Detailed Explanation of the Python Script

This document provides a comprehensive, block-wise explanation of the provided Python code. It describes the purpose of each code section, the logic behind various operations, and the workflow used to produce the final 5C consolidated report.

---

1. Importing Libraries and Initial Setup

```
import pandas as pd
from openpyxl import load_workbook
import warnings
import time
warnings.simplefilter('ignore')

start = time.time()
```

Explanation

- **pandas**: Used for data manipulation and analysis.
- **openpyxl**: Used for reading and writing Excel files.
- **warnings**: Used to suppress unnecessary warnings for a cleaner output.
- **time**: Used to measure execution duration.
- The script begins recording execution time.

---

2. File Loading Function

```
def load_file(file_name:str, row_skip:int = 0, foot_skip:int = 0):
    file_directory ='F:/bbnl 5c/'
    data = pd.read_csv(file_directory+file_name, skiprows=row_skip,
skipfooter=foot_skip, encoding='latin1')
    print("Loaded File: ", file_name)
    return data
```

Explanation

This custom function loads CSV files with optional parameters: - `row_skip`: Number of header rows to skip. - `foot_skip`: Number of footer rows to skip. - **latin-1 encoding**: Used to avoid UTF-8 errors.

This function is used repeatedly to load all the required input datasets.

---

3. Loading All Required Input Files

Eight input datasets are loaded using the `load_file()` function.

```
five = load_file('5c.csv', 2)
amc = load_file('lms.csv', 3, 3)
ava = load_file('prev day ava.csv', 2, 2)
it = load_file('itpc.csv')
bb = load_file('bbnw.csv')
prev_day_1 = load_file('prev day 1.csv', 3, 3)
prev_day_2 = load_file('prev day 2.csv', 3, 3)
org = load_file('original.csv', 3, 3)
udy = load_file('udyami.csv', 3, 3)
```

Explanation

These contain: - **5C GP-level data** - **LMS AMC start records** - **Availability data** - **ITPC Mini OLT records** - **BBNW NMS reachability data** - **Previous day GP up count** - **Udyami mini-OLT UPS data**, etc.

---

4. Creating Unique Key (Concat Columns)

Concatenated string keys are created to enable merging across datasets.

```
five['Concat'] = five['State'] + five['District'] + five['Block'] +
five['Panchayat'] + five['GP Location'] + five['GP Location Code']
```

```
amc['Concat'] = amc['State'] + amc['District'] + amc['Block'] + amc['GP'] +
amc['Location Name'] + amc['Loc Code']
```

```
ava['PANCHAYAT'] = ava['PANCHAYAT'].fillna('--')
ava['Concat'] = ava['STATE'] + ava['DISTRICT'] + ava['BLOCK'] +
ava['PANCHAYAT'] + ava['ONT LOCATION NAME'] + ava['LOCATION CODE']
```

Explanation

Since GP-level records across files do not share a common ID, a synthetic key is created by concatenating key fields.

---

5. Merging LMS AMC Dates & Availability Into 5C File

```
merged = pd.merge(five, amc[['Concat', 'AMC2 Start Date']], on='Concat',
how='left')
merged = pd.merge(merged, ava[['Concat', 'ONT AVAILABILITY']], on='Concat',
how='left')
```

Explanation

- LMS AMC start data and previous-day availability data are merged into the 5C dataset.
- Some unnecessary columns are dropped.

AMC date corrections for A&N are applied where missing.

---

6. Fiber Fault Count Calculation

A fiber fault count is computed by checking fiber fault columns.

```
def fiber_fault_count(row):
    return 5 - sum([... == "--"])
```

Explanation

Each GP has 5 possible fiber fault types. A value of `--` means "no fault". Fault count = total fault types − count of `--` values.

Two subsets are created: - **merged_1** → Phase-1 GPs - **merged_2** → Phase-2 GPs

---

7. Categorizing GPs Based on AMC Date

Four categories created per phase: - In AMC - Not in AMC

```
merged_1_inamc = ...
merged_1_notinamc = ...
```

Explanation

GPs with missing AMC date are considered **not in AMC**.

---

8. State-wise GP Count Table

```
final_1 = merged_1['State'].value_counts().reset_index()
```

Explanation

This forms the base table for the final report. Totals are appended.

---

9. UP + UNKNOWN_PREVIOUSLY_UP Table

Using pivot tables:

```
up_gps_1 = pd.pivot_table(... sum(x == 'UP'))
```

Explanation

Counts GPs that were UP or UP earlier. Added to main table.

---

10. No-AMC Down GPs (No Fiber Fault)

Pivot table calculates: - No fiber faults - Various categories of fiber TT IDs

Results merged into main table.

---

11. AMC Started as per LMS

AMC dataset is merged with Phase info and counted. Missing AMC dates for A&N are corrected. Counts merged into main table.

---

12. AMC Fiber Faults – Detailed Fault Categories

Pivot tables compute counts for: - ONT Faulty/Missing - CCU Faulty/Missing - SPV Issues - Power/ Electricity issues - Custodian issues - Earthing issues, etc.

All these become columns in the final table.

---

13. AMC Down but No Fiber Fault GPs

Separate pivot calculates GPs that are: - DOWN - Have 0 fiber faults

Added as new column.

---

14. Mini OLT Total Count & UP Count

Mini-OLT data (ITPC file + BBNW reachability file) is merged. State-wise: - Total Mini OLTs - UP Mini OLTs

are computed.

---

15. Mini OLT Based Recovery (Udyami File)

State-wise counts of GPs that are DOWN but UP due to Mini OLT.

Added to main table.

---

16. Previous Day UP Counts

Merged directly:

```
final_1 = pd.merge(final_1, prev_day_1)
```

---

17. Availability > 98% (High Availability GPs)

Ont availability ≥98% counted per state and added.

---

18. Derived Columns

Several calculated fields: - AMC Started − Inventory NA - Leased + Lossy - ONT Faulty + Missing

---

19. Re-ordering Final Columns

Both final_1 and final_2 DataFrames are arranged in a structured sequence.

---

20. Writing Data to Existing Excel Template

Phase-1 Writing

```
wb = load_workbook(...)
ws = wb.active
```

For each state in `final_1`, the matching state row in Excel (row 2 onwards) is updated.

Phase-2 Writing

Phase-2 rows start from row 50.

File Saved

Both updates are saved back into the template.

---

21. Execution Time Printed

At the end:

```
print("Time Taken:", round(end - start, 2), "Seconds")
```

Summary of Workflow

1. Load raw datasets.
2. Standardize and merge them using a unique concatenated key.
3. Compute fiber faults and GP status categories.
4. Prepare state-wise summary tables for:
5. GP visibility
6. UP/Down statistics
7. AMC, fiber fault & equipment fault info
8. Mini OLT stats
9. Previous day availability and UP counts
10. Merge all into final summary tables for Phase-1 and Phase-2.
11. Write results into the predefined Excel report template.

If you want, I can also create: - A flowchart of the entire logic - Visual tables explaining each computed metric - A PDF or Word version of this documentation