PharmaSUG China 2026 - Paper PO183

DupChecker: A Python Tool for Detecting Code Duplication in Double Programming in Clinical Trials

Xianhua Zeng, Taimei Intelligence Pharmaceutical, Shanghai, China

ABSTRACT

In the clinical trial industry, double programming is a key methodology to ensure data accuracy and reliability. However, code duplication between main and QC programming teams undermines the independence required in dual programming. DupChecker is a Python-based desktop application that identifies and reports code duplication between main and QC SAS programs. This paper introduces the tool's functionality, its implementation, and its application in the clinical trial industry.

The application is available for download on my GitHub:

https://github.com/XianhuaZeng/Projects/raw/master/DupChecker/DupChecker.exe

INTRODUCTION

Double programming in clinical trials requires independent teams to write separate SAS programs for generating outputs such as tables, listings, and figures (TLF). The independence of these programs ensures unbiased verification of clinical trial data. However, code duplication—whether intentional or unintentional—can compromise this process, leading to potential regulatory concerns and reduced credibility.

To address this, DupChecker was developed as a Python-based tool that uses a sliding-window algorithm for detecting code duplication. The algorithm first filters out comment lines (both block comments delimited by /* and */ and line comments beginning with *), then normalises each remaining line by collapsing whitespace, optionally removing curly braces, and converting the entire line to lower case. It then identifies blocks of consecutive matching lines that appear in both the main and QC programs using a strictly aligned sliding-window approach, which ensures that only windows advancing in sync on both sides are merged into a single duplicate block. Robust encoding detection (UTF-8 and GBK) ensures correct handling of files containing Chinese characters. By automating the comparison process and generating comprehensive Excel reports sorted by the number of duplicate lines in descending order, DupChecker provides users with a simple yet powerful way to uphold double programming standards.

Below is the GUI of DupChecker.

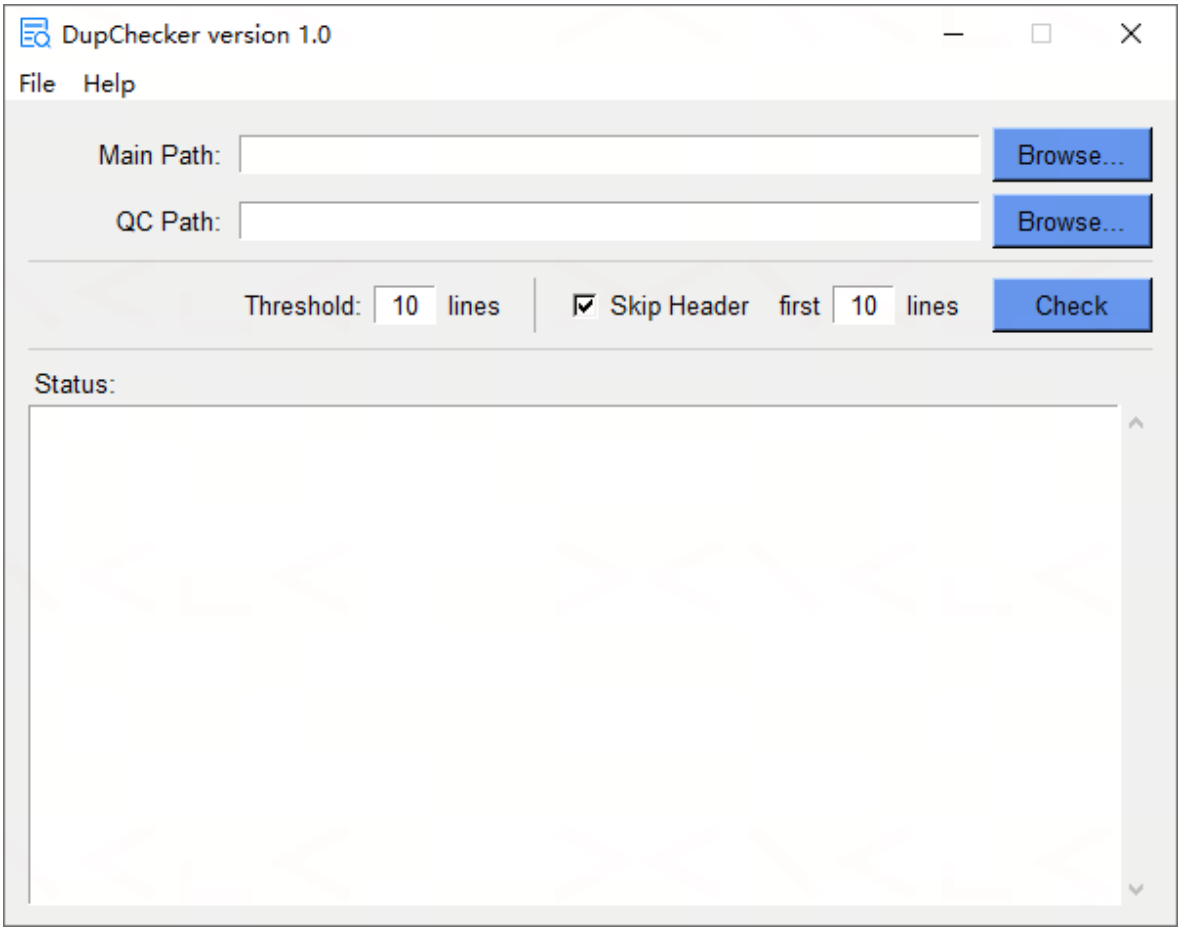


Figure 1. GUI of DupChecker

FUNCTIONALITIES

The DupChecker GUI is designed to be user-friendly and efficient (Figure 1). The tool's core functionalities include:

1. Main Path and QC Path Selection: Users can browse and specify directories containing SAS programs from the main and QC teams.

2. Duplicate Detection: DupChecker uses a sliding-window algorithm with comment filtering, case-insensitive normalisation, and strict aligned merging to accurately detect duplicate code blocks between main and QC programs.

3. Report Generation: DupChecker generates contains details of the detected duplications, including the number of duplicate lines, file names (with hyperlinks for direct access to the problematic files), page ranges, and a preview of the duplicated code.

4. Real-Time Feedback: DupChecker provides status updates during the comparison process, keeping users informed of progress and results.

KEY FEATURES

1. Sliding-Window Detection: The core algorithm uses a strictly aligned sliding-window approach with comment filtering (/* */ and * style), case-insensitive line normalisation, and automatic UTF-8/GBK encoding detection for reliable results across different file encodings.

2. Customizable Parameters: Users can adjust the threshold (minimum number of matching lines) and choose whether to skip header lines, providing flexible control over duplication detection. Results are automatically sorted by duplicate line count in descending order so the most significant duplications appear first.

3. Cross-Platform Compatibility: DupChecker supports various versions of Windows and is lightweight for easy deployment.

4. User-Friendly Interface: Designed for non-programmers, the tool abstracts complex operations into a simple GUI.

TESTING

To run the DupChecker function:

1. Open DupChecker. In the Options bar, set the Threshold value (minimum number of matching lines for duplicate detection). The default value is 10. Optionally enable Skip Header and specify how many header lines to skip.

2. Use the "Browse..." buttons to select the folders containing the main and QC SAS programs.

3. Click the "Check" button to initiate the process.

4. DupChecker will run the duplicate detection algorithm to perform the code duplication analysis..

If everything went well, you should see the GUI looks like below:

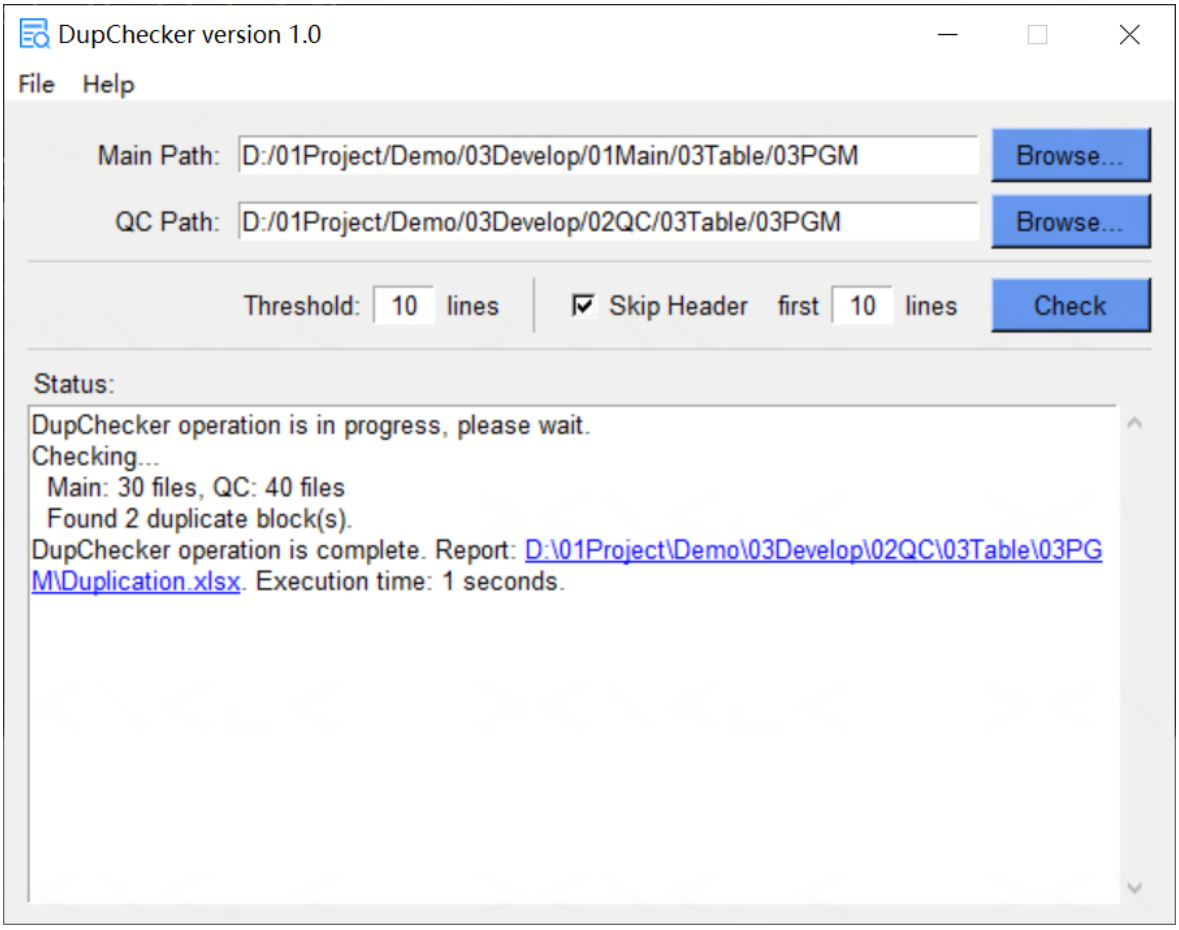
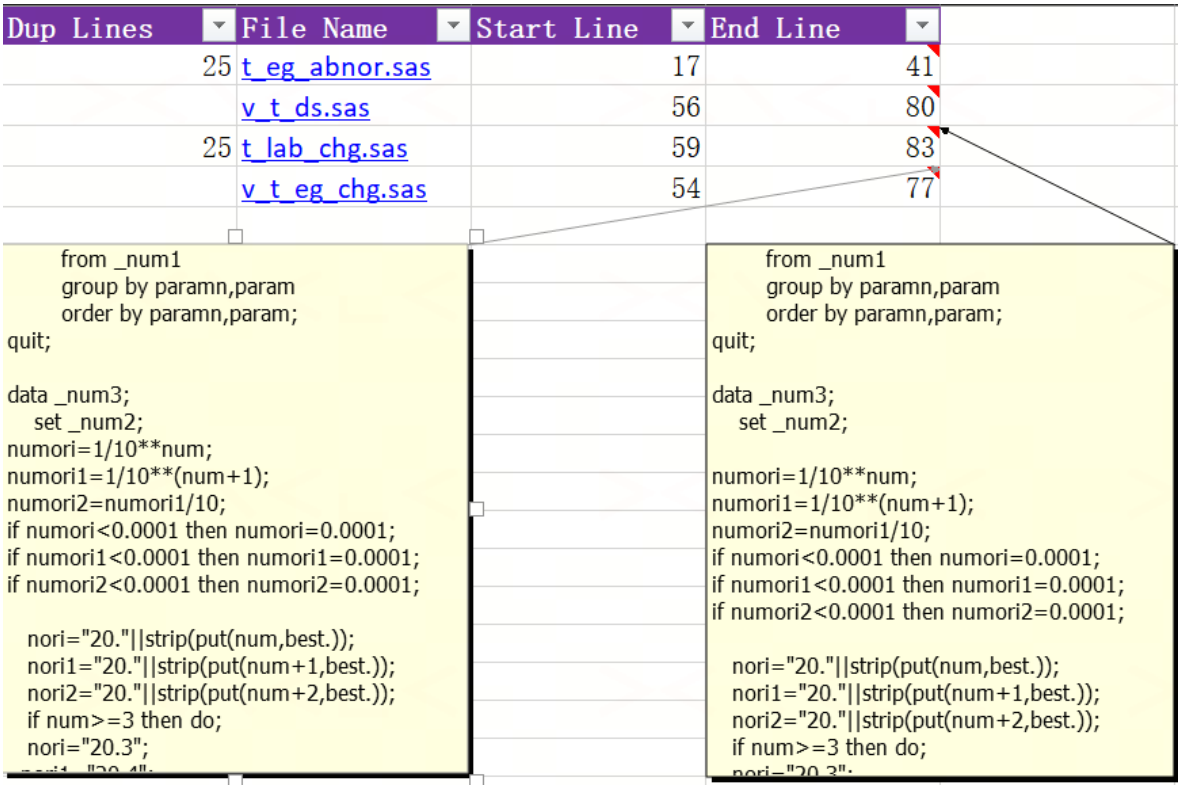


Figure 2. Execution Status of DupChecker

Once the operation is complete, a message will appear in the "Status" box, indicating the location of the generated issue report (Duplication.xlsx) and the total execution time.

The report is saved as an Excel file (Duplication.xlsx), which contains details of the detected duplications, including the number of duplicate lines, file names (with hyperlinks for direct access to the problematic files), page ranges, and a preview of the duplicated code. Below is an example of the output:



Dup Lines	File Name	Start Line	End Line
25	t_eg_abnor.sas	17	41
	v_t_ds.sas	56	80
25	t_lab_chg.sas	59	83
	v_t_eg_chg.sas	54	77

```
      from _num1
      group by paramn,param
      order by paramn,param;
quit;

data _num3;
   set _num2;
numori=1/10**num;
numori1=1/10**(num+1);
numori2=numori1/10;
if numori<0.0001 then numori=0.0001;
if numori1<0.0001 then numori1=0.0001;
if numori2<0.0001 then numori2=0.0001;

  nori="20."||strip(put(num,best.));
  nori1="20."||strip(put(num+1,best.));
  nori2="20."||strip(put(num+2,best.));
  if num>=3 then do;
  nori="20.3";
```

```
      from _num1
      group by paramn,param
      order by paramn,param;
quit;

data _num3;
   set _num2;

numori=1/10**num;
numori1=1/10**(num+1);
numori2=numori1/10;
if numori<0.0001 then numori=0.0001;
if numori1<0.0001 then numori1=0.0001;
if numori2<0.0001 then numori2=0.0001;

  nori="20."||strip(put(num,best.));
  nori1="20."||strip(put(num+1,best.));
  nori2="20."||strip(put(num+2,best.));
  if num>=3 then do;
```

Figure 3. Duplication **File**

CONCLUSION

DupChecker is an innovative Python tool for maintaining the integrity of double programming in clinical trials. By automating the detection of code duplication, it ensures the independence of main and QC programming teams, safeguarding the quality and reliability of clinical trial outputs.

REFERENCE

Fredrik Lundh. “Tkinter”. Available at https://docs.python.org/3/library/tk.html

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Xianhua Zeng
Enterprise: Taimei Intelligence Pharmaceutical
Address: 6th Floor, Building 24, Phase 3, Caohejing Technology Oasis, No. 1999 Yishan Road, Minhang District, Shanghai, China, 200233
E-mail: huazizeng@gmail.com
Web: https://www.taimei.com/
http://www.xianhuazeng.com/