convert_scripts_to_pdf

```
import os
from fpdf import FPDF
import argparse


def convert_scripts_to_pdf(do_file_path, pdf, encoding="utf-8"):
    # Read the contents of the .do file
    with open(do_file_path, 'r', encoding=encoding) as do_file:
        do_content = do_file.read()

    # Add a new page with the .do file title
    pdf.add_page()
    pdf.set_font('Arial', 'B', 10)
    pdf.cell(0, 10, os.path.basename(do_file_path), ln=True)

    # Add the .do file content to the PDF
    pdf.set_font('Courier', '', 8)
    pdf.multi_cell(0, 4, do_content.replace('\n\n', '\n').strip())
    pdf.ln()


def process_folder(folder_path, output_pdf_path, extension='do'):
    # Initialize the PDF document
    pdf = FPDF()

    # Walk through the folder recursively
    for root, dirs, files in os.walk(folder_path):
        for file in files:
            # Check if the file is a Stata .do file
            if file.endswith(f'.{extension}'):
                do_file_path = os.path.join(root, file)

                # Convert the .do file to PDF
                convert_scripts_to_pdf(do_file_path, pdf, encoding="latin1")

                print(f"Processed {do_file_path}")

    # Save the PDF file
    pdf.output(output_pdf_path)


if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Convert scripts to PDF')
    parser.add_argument('folder_path', type=str, help='Path to the folder containing scripts')
    parser.add_argument('output_pdf_path', type=str, help='Path to the output PDF file')
    parser.add_argument('--ext', type=str, default='py', help='File extension to search for (default is "py")')
    args = parser.parse_args()

    # Call the function to process the folder
    process_folder(args.folder_path, args.output_pdf_path, extension=args.ext)
```

convert_scripts_to_pdf_highlight

```
import os
from pathlib import Path
import pdfkit
import PyPDF2
from pygments import highlight
from pygments.lexers import get_lexer_by_name
from pygments.formatters import HtmlFormatter
import argparse

def convert_scripts_to_pdf(do_file_path, output_pdf_path,
                    encoding="utf-8", language='stata'):
    # Read the contents of the .do file
    with open(do_file_path, 'r', encoding=encoding) as do_file:
        do_content = do_file.read()

    # Highlight the .do file content using Pygments
    lexer = get_lexer_by_name(language)
    formatter = HtmlFormatter(style='default')
    highlighted_code = highlight(do_content, lexer, formatter)

    # Generate the HTML content
    html_content = f"""
    <html>
    <head>
        <style>
            {formatter.get_style_defs('.highlight')}
        </style>
    </head>
    <body>
        <p><h1>{Path(do_file_path).stem}</h1></p>
        <pre class="highlight">{highlighted_code}</pre>
    </body>
    </html>
    """

    # Convert HTML to PDF using pdfkit
    pdfkit.from_string(html_content, output_pdf_path)

def merge_pdfs(input_pdf_paths, output_pdf_path):
    merger = PyPDF2.PdfMerger()

    # Merge all input PDFs into a single PDF
    for pdf_path in input_pdf_paths:
        merger.append(pdf_path)

    # Save the merged PDF to the output path
    merger.write(output_pdf_path)
    merger.close()

    for file in input_pdf_paths:
        file_path = Path(file)
        os.remove(file_path)

def process_folder(folder_path, output_pdf_path,
                   extension='do', language='stata'):
    # Walk through the folder recursively
    pdf_files_paths = []
    for root, dirs, files in os.walk(folder_path):
        for file in files:
            # Check if the file is a Stata .do file
            if file.endswith(f'.{extension}'):
                do_file_path = os.path.join(root, file)
                pdf_file_path = f"./{folder_path}/{file.strip('.{extension}')}.pdf"

                # Convert the .do file to PDF
                print("###############", pdf_file_path)
                convert_scripts_to_pdf(do_file_path, pdf_file_path,
                                       encoding="latin1", language=language)
                pdf_files_paths.append(pdf_file_path)
                print(f"Converted {do_file_path} to {pdf_file_path}")

    # Merge all the PDF files into a single PDF using pdfkit
    merge_pdfs(pdf_files_paths, output_pdf_path)

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Convert scripts files to PDF')
    parser.add_argument('folder_path', type=str, help='Path to the folder containing scripts')
    parser.add_argument('output_pdf_path', type=str, help='Path to the output PDF file')
    parser.add_argument('--ext', type=str, default='py', help='File extension to search for (default is "py")')
    parser.add_argument('--lang', type=str, default='python', help='Language to use for syntax highlighting (default is "python")')
    args = parser.parse_args()

    # Call the function to process the folder
    process_folder(args.folder_path, args.output_pdf_path,
                   extension=args.ext, language=args.lang)
```