**Conversation with Gemini**

SH icon

batchstitch.sh

SH

See the bash script.

The problem is, it processes files with PART 1, PART 10, PART 11, etc instead of PART 1, PART 2, PART 3, etc. So basically it treats digits as characters and not numbers. File Names are like: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 1   श्रीमद भगवत महा पुराण संस्कृत में.opus, SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 2   श्रीमद भगवत महा पुराण संस्कृत में.opus, SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 3   श्रीमद भगवत महा पुराण संस्कृत में.opus .. and so on up to 30 parts. PART 30 has file name as SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 30 LAST   श्रीमद भगवत महा पुराण संस्कृत में.opus.

As you can see, digits after PART changes other wise file names are same. Can you modify script so that it processes File names with PART 1, PART 2 in that order?

Analysis

Analysis

* Query successful

To ensure the files are processed in numerical order (PART 1, PART 2, ..., PART 30), you need to modify the patterns array within the loop to prioritize two-digit part numbers with a leading zero (e.g., PART 01, PART 02) and then single-digit numbers. This ensures that "PART 1" is matched before "PART 10", and so on. Also, the current script is already using printf "%02d" when constructing temp\_name, which helps with alphabetical sorting of the temporary files.

Here's the modified section of the script:

Bash

# Try to find files for parts 1-30 using pattern matching

for i in {1..30}; do

# Try different patterns for each part number, prioritizing 0-padded numbers

patterns=(

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\_$(printf "%02d" $i)\*[^0-9]\*श्रीमद\*.opus" # e.g., PART\_01, PART\_02

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\_$i\*[^0-9]\*श्रीमद\*.opus" # e.g., PART\_1, PART\_2 (for single digits)

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\_$i[^0-9]\*श्रीमद\*.opus" # Original pattern for robustness

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\*0$i[^0-9]\*श्रीमद\*.opus"

)

found=false

for pattern in "${patterns[@]}"; do

# Use find with pattern matching

while IFS= read -r -d '' file; do

if [[ -f "$file" ]]; then

echo "Found PART $i: $(basename "$file")"

# Ensure the temporary file name is always 0-padded for correct sorting

temp\_name="bhagwat\_part\_$(printf "%02d" $i).opus"

temp\_path="$TEMP\_DIR/$temp\_name"

cp "$file" "$temp\_path" || {

echo "Error: Failed to copy PART $i"

continue

}

# Convert MSYS2 path to Windows path for FFmpeg

win\_temp\_path=$(cygpath -w "$temp\_path" 2>/dev/null || echo "$temp\_path")

# Escape backslashes and wrap in quotes for the file list

echo "file '${win\_temp\_path//\\/\/}'" >> "$TEMP\_LIST"

found\_files[$i]="$file"

found=true

break

fi

done < <(find . -maxdepth 1 -name "$pattern" -print0 2>/dev/null)

if $found; then

break

fi

done

if ! $found; then

echo "Warning: PART $i not found with any pattern"

fi

done

**Reasoning for the change:**

The key to processing in numerical order is to ensure that when the script looks for files, it prioritizes patterns that match the numerical order you desire.

1. **\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\_$(printf "%02d" $i)\*[^0-9]\*श्रीमद\*.opus**: This new pattern explicitly looks for "PART\_" followed by a *two-digit* number (e.g., "PART\_01", "PART\_02", ..., "PART\_10", "PART\_11"). This ensures that single-digit parts are matched as "01", "02", etc., which naturally sorts correctly before "10", "11", etc., when sorted alphabetically.
2. **\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\_$i\*[^0-9]\*श्रीमद\*.opus**: This pattern handles cases where the part number might not be zero-padded (e.g., "PART 1", "PART 2"). By putting this after the zero-padded pattern, it acts as a fallback.
3. The remaining patterns from your original script are kept for robustness in case filenames have slight variations.

By prioritizing the PART\_0N format in the patterns array and ensuring the temp\_name is consistently zero-padded with printf "%02d" $i, the temporary files created in $TEMP\_DIR will have names like bhagwat\_part\_01.opus, bhagwat\_part\_02.opus, ..., bhagwat\_part\_10.opus, etc. When ffmpeg reads TEMP\_LIST, these file paths, being alphabetically sorted due to the 0-padding, will lead to the correct numerical concatenation order.

Your code failed:

$ ./batchstitich\_modified.sh

Warning: PART 1 not found with any pattern

Warning: PART 2 not found with any pattern

Warning: PART 3 not found with any pattern

Warning: PART 4 not found with any pattern

Warning: PART 5 not found with any pattern

Warning: PART 6 not found with any pattern

Warning: PART 7 not found with any pattern

Warning: PART 8 not found with any pattern

Warning: PART 9 not found with any pattern

Warning: PART 10 not found with any pattern

Warning: PART 11 not found with any pattern

Warning: PART 12 not found with any pattern

Warning: PART 13 not found with any pattern

Warning: PART 14 not found with any pattern

Warning: PART 15 not found with any pattern

Warning: PART 16 not found with any pattern

Warning: PART 17 not found with any pattern

Warning: PART 18 not found with any pattern

Warning: PART 19 not found with any pattern

Warning: PART 20 not found with any pattern

Warning: PART 21 not found with any pattern

Warning: PART 22 not found with any pattern

Warning: PART 23 not found with any pattern

Warning: PART 24 not found with any pattern

Warning: PART 25 not found with any pattern

Warning: PART 26 not found with any pattern

Warning: PART 27 not found with any pattern

Warning: PART 28 not found with any pattern

Warning: PART 29 not found with any pattern

Warning: PART 30 not found with any pattern

Show thinking

The issue is likely due to the inclusion of \_ (underscore) in the first pattern PART\_$(printf "%02d" $i). Your filenames have a space, not an underscore, before the part number (e.g., PART 1, PART 10).

Let's adjust the patterns to correctly match the space and ensure the most specific pattern is tried first for numerical ordering. The key is to account for the space and to try patterns that ensure "PART 1" is found before "PART 10".

Here's the corrected and refined patterns array:

Bash

# Try to find files for parts 1-30 using pattern matching

for i in {1..30}; do

# Try different patterns for each part number, prioritizing two-digit and then single-digit numbers

patterns=(

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART $(printf "%02d" $i)\*[^0-9]\*श्रीमद\*.opus" # Matches "PART 01", "PART 02", ..., "PART 10", etc.

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART $i[^0-9]\*श्रीमद\*.opus" # Matches "PART 1", "PART 2", etc.

"\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\*$i[^0-9]\*श्रीमद\*.opus" # Original, more general pattern as a fallback

)

found=false

for pattern in "${patterns[@]}"; do

# Use find with pattern matching

# The -print0 and read -r -d '' handles filenames with spaces correctly.

while IFS= read -r -d '' file; do

if [[ -f "$file" ]]; then

echo "Found PART $i: $(basename "$file")"

# Ensure the temporary file name is always 0-padded for correct sorting

temp\_name="bhagwat\_part\_$(printf "%02d" $i).opus"

temp\_path="$TEMP\_DIR/$temp\_name"

cp "$file" "$temp\_path" || {

echo "Error: Failed to copy PART $i"

continue

}

# Convert MSYS2 path to Windows path for FFmpeg

win\_temp\_path=$(cygpath -w "$temp\_path" 2>/dev/null || echo "$temp\_path")

# Escape backslashes and wrap in quotes for the file list

echo "file '${win\_temp\_path//\\/\/}'" >> "$TEMP\_LIST"

found\_files[$i]="$file"

found=true

break # Break from inner while loop once a file is found for this part

fi

done < <(find . -maxdepth 1 -name "$pattern" -print0 2>/dev/null)

if $found; then

break # Break from outer for loop (patterns) once a file is found for this part

fi

done

if ! $found; then

echo "Warning: PART $i not found with any pattern"

fi

done

**Changes and Reasoning:**

1. **PART $(printf "%02d" $i)**: This is the most crucial change. It now explicitly matches "PART " followed by a zero-padded two-digit number (e.g., "PART 01", "PART 02", "PART 10"). This ensures that single-digit parts are found as "01", "02", etc., which when combined with printf "%02d" for the temporary file name, ensures natural alphabetical sorting (and thus numerical order) for ffmpeg.
2. **PART $i**: This pattern is for cases where the part number is a single digit without a leading zero (e.g., "PART 1", "PART 2"). This is placed after the zero-padded pattern so that if both "PART 01" and "PART 1" existed, "PART 01" would be preferred.
3. **Order of Patterns**: The order of patterns in the patterns array is important. We want to prioritize the most specific and numerically "correct" pattern first.
   * "\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART $(printf "%02d" $i)\*[^0-9]\*श्रीमद\*.opus": This will correctly match PART 01, PART 02, etc., and PART 10, PART 11, etc.
   * "\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART $i[^0-9]\*श्रीमद\*.opus": This will then catch PART 1, PART 2, etc., if they are not zero-padded.
   * The last pattern "\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*PART\*$i[^0-9]\*श्रीमद\*.opus" is a more general fallback.

With these adjustments, the script should now correctly identify and process the files in the desired numerical order. Here's the corrected batchstitch.sh script to address the issue of files not being found due to incorrect patterns. The primary change is in the patterns array, which now accurately reflects the spaces and specific keywords in your filenames, such as "PART " (with a space) instead of "PART\_" (with an underscore), and the "LAST" keyword for Part 30.

This script should now correctly identify and process all 30 parts of your OPUS files.

Bash

#!/bin/bash

# Script to rename and stitch multiple OPUS files into one

# For MSYS2 environment - handles Unicode filenames with Windows path compatibility

# Set the directory path (Windows path converted to MSYS2 format)

SOURCE\_DIR="/d/dump/Religious Books PDF And Audio"

OUTPUT\_FILE="/d/dump/Religious Books PDF And Audio/SHRIMAD\_BHAGWAT\_MAHA\_PURAN\_COMPLETE\_SANSKRIT\_FULL.opus"

TEMP\_LIST="/d/dump/Religious Books PDF And Audio/opus\_files\_list.txt"

TEMP\_DIR="/d/dump/Religious Books PDF And Audio/bhagwat\_temp"

echo "=== OPUS File Stitching Script with Flexible Matching ==="

echo "Source Directory: $SOURCE\_DIR"

echo "Output File: $OUTPUT\_FILE"

echo

# Change to source directory

cd "$SOURCE\_DIR" || {

echo "Error: Cannot access directory '$SOURCE\_DIR'"

exit 1

}

# Create temporary directory for renamed files

mkdir -p "$TEMP\_DIR"

echo "Step 1: Scanning for OPUS files..."

echo "Available OPUS files with 'SHRIMAD BHAGWAT' in name:"

echo "----------------------------------------"

# List all matching files first to see what we have

ls -1 \*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*.opus 2>/dev/null | head -10

echo "..."

echo

# Create temporary file list

> "$TEMP\_LIST"

declare -A found\_files # Associative array to store paths of found files

# Try to find files for parts 1-30 using pattern matching

for i in {1..30}; do

declare -a patterns # Declare as array for safety

if [ "$i" -eq 30 ]; then

# Specific pattern for PART 30 with "LAST"

patterns=(

"\*SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 30 LAST श्रीमद भगवत महा पुराण संस्कृत में.opus"

)

else

# General pattern for other parts (1-29)

patterns=(

"\*SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART $i श्रीमद भगवत महा पुराण संस्कृत में.opus"

# Add a more general pattern if the above is too strict, e.g., for slight variations

"\*SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART $i\*श्रीमद\*.opus"

)

fi

found=false

for pattern in "${patterns[@]}"; do

# Use find with pattern matching. -print0 and read -d '' are crucial for filenames with spaces/special chars.

while IFS= read -r -d '' file; do

if [[ -f "$file" ]]; then

echo "Found PART $i: $(basename "$file")"

# Ensure the temporary file name is always 0-padded for correct sorting

temp\_name="bhagwat\_part\_$(printf "%02d" $i).opus"

temp\_path="$TEMP\_DIR/$temp\_name"

cp "$file" "$temp\_path" || {

echo "Error: Failed to copy PART $i"

continue

}

# Convert MSYS2 path to Windows path for FFmpeg

win\_temp\_path=$(cygpath -w "$temp\_path" 2>/dev/null || echo "$temp\_path")

# Escape backslashes and wrap in quotes for the file list

echo "file '${win\_temp\_path//\\/\/}'" >> "$TEMP\_LIST"

found\_files[$i]="$file"

found=true

break # Found the file for this part, move to next i

fi

done < <(find . -maxdepth 1 -name "$pattern" -print0 2>/dev/null)

if $found; then

break # Found the file for this part with one of the patterns, move to next i

fi

done

if ! $found; then

echo "Warning: PART $i not found with any pattern"

fi

done

if [ ${#found\_files[@]} -eq 0 ]; then

echo "❌ ERROR: No OPUS files found matching the patterns. Exiting."

rm -rf "$TEMP\_DIR"

rm -f "$TEMP\_LIST"

exit 1

fi

echo

echo "Step 2: Temporary files copied and list created."

echo "Temporary directory: $TEMP\_DIR"

echo "Temporary file list: $TEMP\_LIST"

echo "Files to be concatenated (first 5 lines):"

head -n 5 "$TEMP\_LIST"

echo "..."

echo

echo "Step 3: Concatenating files using FFmpeg..."

# Convert MSYS2 paths in TEMP\_LIST to Windows paths for FFmpeg

# Using sed to replace /d/ with d:/ (simplistic, might need refinement for complex paths)

# and converting forward slashes to backslashes

win\_temp\_list="${TEMP\_LIST}.win"

# This sed command is more robust for MSYS2 path conversion

sed 's|/mnt/\([a-zA-Z]\)|\1:/|; s|/\([a-zA-Z]\)|\1:/|; s|/|\\|g' "$TEMP\_LIST" > "$win\_temp\_list"

# Convert the final output path to Windows format

win\_output\_file=$(cygpath -w "$OUTPUT\_FILE" 2>/dev/null || echo "$OUTPUT\_FILE")

# Use ffmpeg to concatenate

if ffmpeg -f concat -safe 0 -i "$win\_temp\_list" -c copy "$win\_output\_file"; then

echo

echo "✅ SUCCESS! Files concatenated successfully."

echo "Output file: $OUTPUT\_FILE"

# Show file size

if [[ -f "$OUTPUT\_FILE" ]]; then

file\_size=$(du -h "$OUTPUT\_FILE" | cut -f1)

echo "File size: $file\_size"

# Get duration using ffprobe if available

if command -v ffprobe >/dev/null 2>&1; then

duration=$(ffprobe -v quiet -show\_entries format=duration -of csv="p=0" "$win\_output\_file" 2>/dev/null)

if [[ -n "$duration" ]]; then

# Convert seconds to hours:minutes:seconds

hours=$((${duration%.\*} / 3600))

minutes=$(((${duration%.\*} % 3600) / 60))

seconds=$((${duration%.\*} % 60))

echo "Duration: ${hours}h ${minutes}m ${seconds}s"

fi

fi

fi

else

echo

echo "❌ ERROR: Concatenation failed!"

echo "Check FFmpeg output above for specific error details."

rm -rf "$TEMP\_DIR"

rm -f "$TEMP\_LIST"

exit 1

fi

# Clean up temporary files and directory

echo

echo "Step 4: Cleaning up temporary files..."

rm -rf "$TEMP\_DIR"

rm -f "$TEMP\_LIST"

rm -f "$win\_temp\_list" # Remove the Windows-formatted list as well

echo "Temporary files cleaned up."

Again failed.

j1304@dhawal123 MSYS /d/dump/Religious Books PDF And Audio

$ ./batchstitich\_modified.sh

Found PART 1: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 1   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 2: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 2   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 3: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 3   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 4: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 4   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 5: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 5   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 6: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 6   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 7: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 7   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 8: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 8   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 9: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 9   श्रीमद भगवत महा पुराण संस्कृत में.opus

./batchstitich\_modified.sh: line 29: : No such file or directory

Found PART 10: SHRIMAD BHAGWAT MAHA PURAN COMPLETE IN SANSKRIT PART 10   श्रीमद भगवत महा पुराण संस्कृत में.opus

Issue is Asian Language devnagri (hindi) characters in the file. You should first clean those characters and rename the files as is done in my original script. Than process like PART 1, PART 2, PART 3 in that order.

You are absolutely right! My apologies. I missed the crucial part of your original script where you were renaming the files to remove the Devanagari characters and standardize the names *before* copying them. The cp command was failing because the file variable still contained the problematic characters, which cp (or the underlying Windows filesystem via MSYS2) couldn't handle directly as a source path in some contexts.

The No such file or directory error at line 29 (which corresponds to cp "$file" "$temp\_path") is a classic symptom of an invalid or unresolvable filename.

We need to re-introduce the new\_file\_name cleaning logic and ensure cp uses that cleaned name, then the find command can be simpler. The best approach is to clean the filenames *first*, then match the cleaned names.

Here's the corrected and more robust approach:

1. **Rename/Clean Files First:** Before looping through parts, we'll perform a one-time operation to rename all problematic files into a standardized, ASCII-only format. This makes subsequent find and cp operations much more reliable.
2. **Simplified Patterns:** With cleaned filenames, the patterns to match PART 1, PART 2, etc., become straightforward.
3. **Numerical Sorting:** We'll still use the printf "%02d" for temporary file names to ensure correct numerical order during concatenation.

Here's the modified script:

Bash

#!/bin/bash

# Script to rename and stitch multiple OPUS files into one

# For MSYS2 environment - handles Unicode filenames with Windows path compatibility

# Set the directory path (Windows path converted to MSYS2 format)

SOURCE\_DIR="/d/dump/Religious Books PDF And Audio"

OUTPUT\_FILE="/d/dump/Religious Books PDF And Audio/SHRIMAD\_BHAGWAT\_MAHA\_PURAN\_COMPLETE\_SANSKRIT\_FULL.opus"

TEMP\_LIST="/d/dump/Religious Books PDF And Audio/opus\_files\_list.txt"

TEMP\_DIR="/d/dump/Religious Books PDF And Audio/bhagwat\_temp"

echo "=== OPUS File Stitching Script with Flexible Matching ==="

echo "Source Directory: $SOURCE\_DIR"

echo "Output File: $OUTPUT\_FILE"

echo

# Change to source directory

cd "$SOURCE\_DIR" || {

echo "Error: Cannot access directory '$SOURCE\_DIR'"

exit 1

}

# Create temporary directory for renamed files

mkdir -p "$TEMP\_DIR"

echo "Step 1: Cleaning and Renaming files to ASCII-compatible names..."

echo "------------------------------------------------------------"

# Create a temporary directory to hold the ASCII-compatible renamed files

CLEAN\_DIR="$TEMP\_DIR/cleaned\_files"

mkdir -p "$CLEAN\_DIR"

# Find all original OPUS files and rename them into CLEAN\_DIR

# This step ensures that all subsequent operations deal with ASCII-only filenames.

find . -maxdepth 1 -name "\*SHRIMAD\*BHAGWAT\*PURAN\*SANSKRIT\*.opus" -print0 | while IFS= read -r -d '' original\_file; do

# Extract filename without path

filename=$(basename "$original\_file")

# Clean the filename to remove Devanagari characters and normalize spaces

# Replace Hindi characters (and common problematic chars) with nothing,

# then replace multiple spaces with single spaces, and finally trim.

# We use a pattern that matches the specific "श्रीमद भगवत महा पुराण संस्कृत में" part.

cleaned\_filename=$(echo "$filename" | \

sed 's/[[:space:]]\*श्रीमद भगवत महा पुराण संस्कृत में\.opus//' | \

sed 's/[[:space:]]\*श्रीमद भगवत महा पुराण संस्कृत मे\.opus//' | \

sed 's/[[:space:]]\*श्रीमद भगवत महा पुराण संस्क्रित में\.opus//' | \

sed 's/[[:space:]]\*श्रीमद भगवत महा पुराण संस्क्रित मे\.opus//' | \

sed 's/[[:space:]]\*श्रीमद भगवत महा पुराण संस्कृतम्\.opus//' | \

sed 's/[[:space:]]\*श्रीमद भगवत महा पुराण संस्कृतम\.opus//' | \

sed 's/[[:space:]]\*[^\x00-\x7F]\*\.opus/\.opus/' | \

sed 's/[[:space:]][[:space:]]\*/ /g' | \

sed 's/PART[[:space:]]\([0-9]\*\)[[:space:]]\*LAST/PART\_\1\_LAST/' | \

sed 's/PART[[:space:]]\([0-9]\*\)/PART\_\1/' | \

sed 's/ /\_/g' | \

tr -d '\n\r' | \

sed 's/\_\_/\_/g' | \

sed 's/\_-\_/-/g' | \

sed 's/\_-\_/\_/g' | \

sed 's/\.\_/\./' | \

sed 's/\_\.opus/\.opus/')

# Add .opus extension back if somehow lost, and ensure no leading/trailing underscores

cleaned\_filename=$(echo "$cleaned\_filename" | sed 's/\.opus$//' | sed 's/^\_//;s/\_$//')".opus"

# Ensure the part number is separated by an underscore for easier parsing later

cleaned\_filename=$(echo "$cleaned\_filename" | sed 's/PART\([0-9]\)/PART\_\1/g' | sed 's/PART\_\([0-9]\{1,2\}\)\_LAST/PART\_\1\_LAST/g')

# Construct the full path for the cleaned file

cleaned\_file\_path="$CLEAN\_DIR/$cleaned\_filename"

echo "Renaming: $(basename "$original\_file") -> $(basename "$cleaned\_file\_path")"

cp "$original\_file" "$cleaned\_file\_path" || {

echo "Error: Failed to copy '$original\_file' to '$cleaned\_file\_path'"

continue

}

done

echo "------------------------------------------------------------"

echo "Finished cleaning and renaming files."

echo

# Now, change to the CLEAN\_DIR to work with the cleaned filenames

cd "$CLEAN\_DIR" || {

echo "Error: Cannot access cleaned files directory '$CLEAN\_DIR'"

exit 1

}

# Create temporary file list for ffmpeg

> "$TEMP\_LIST"

declare -A found\_files # Associative array to store paths of found files

echo "Step 2: Scanning for cleaned OPUS files and preparing for concatenation..."

echo "----------------------------------------------------------------------"

# Try to find files for parts 1-30 using pattern matching on cleaned names

for i in {1..30}; do

declare -a patterns # Declare as array for safety

if [ "$i" -eq 30 ]; then

# Specific pattern for PART 30 with "LAST" in cleaned names

patterns=(

"\*SHRIMAD\_BHAGWAT\_MAHA\_PURAN\_COMPLETE\_IN\_SANSKRIT\_PART\_${i}\_LAST.opus"

"\*SHRIMAD\_BHAGWAT\_MAHA\_PURAN\_COMPLETE\_IN\_SANSKRIT\_PART\_${i}\_LAST\_.opus" # In case of extra underscore

)

else

# General pattern for other parts (1-29) in cleaned names

patterns=(

"\*SHRIMAD\_BHAGWAT\_MAHA\_PURAN\_COMPLETE\_IN\_SANSKRIT\_PART\_${i}.opus"

"\*SHRIMAD\_BHAGWAT\_MAHA\_PURAN\_COMPLETE\_IN\_SANSKRIT\_PART\_${i}\_.opus" # In case of extra underscore

)

fi

found=false

for pattern in "${patterns[@]}"; do

# Use find with pattern matching. -print0 and read -d '' are crucial for filenames with spaces/special chars.

while IFS= read -r -d '' file; do

if [[ -f "$file" ]]; then

echo "Found PART $i: $(basename "$file")"

# Ensure the temporary file name is always 0-padded for correct sorting

temp\_name="bhagwat\_part\_$(printf "%02d" $i).opus"

temp\_path="$TEMP\_DIR/$temp\_name" # Still copy to the original TEMP\_DIR

# Move the cleaned file to the final temporary directory for processing

mv "$file" "$temp\_path" || { # Use mv instead of cp since we copied to CLEAN\_DIR first

echo "Error: Failed to move PART $i cleaned file to '$temp\_path'"

continue

}

# Convert MSYS2 path to Windows path for FFmpeg

win\_temp\_path=$(cygpath -w "$temp\_path" 2>/dev/null || echo "$temp\_path")

# Escape backslashes and wrap in quotes for the file list

echo "file '${win\_temp\_path//\\/\/}'" >> "$TEMP\_LIST"

found\_files[$i]="$file"

found=true

break # Found the file for this part, move to next i

fi

done < <(find . -maxdepth 1 -name "$pattern" -print0 2>/dev/null)

if $found; then

break # Found the file for this part with one of the patterns, move to next i

fi

done

if ! $found; then

echo "Warning: PART $i not found with any pattern in '$CLEAN\_DIR'"

fi

done

if [ ${#found\_files[@]} -eq 0 ]; then

echo "❌ ERROR: No OPUS files found matching the patterns in the cleaned directory. Exiting."

rm -rf "$TEMP\_DIR"

rm -f "$TEMP\_LIST"

exit 1

fi

echo

echo "Step 3: Temporary files copied and list created."

echo "Temporary directory for final concatenation: $TEMP\_DIR"

echo "Temporary file list: $TEMP\_LIST"

echo "Files to be concatenated (first 5 lines):"

head -n 5 "$TEMP\_LIST"

echo "..."

echo

echo "Step 4: Concatenating files using FFmpeg..."

# Convert MSYS2 paths in TEMP\_LIST to Windows paths for FFmpeg

win\_temp\_list="${TEMP\_LIST}.win"

# This sed command converts /d/ to d:\ and all / to \

# It assumes TEMP\_LIST contains MSYS2 paths relative to the root like /d/dump...

sed 's|^\/\([a-zA-Z]\)\/|\1:\\|; s|/|\\|g' "$TEMP\_LIST" > "$win\_temp\_list"

# Convert the final output path to Windows format

win\_output\_file=$(cygpath -w "$OUTPUT\_FILE" 2>/dev/null || echo "$OUTPUT\_FILE")

# Use ffmpeg to concatenate

if ffmpeg -f concat -safe 0 -i "$win\_temp\_list" -c copy "$win\_output\_file"; then

echo

echo "✅ SUCCESS! Files concatenated successfully."

echo "Output file: $OUTPUT\_FILE"

# Show file size

if [[ -f "$OUTPUT\_FILE" ]]; then

file\_size=$(du -h "$OUTPUT\_FILE" | cut -f1)

echo "File size: $file\_size"

# Get duration using ffprobe if available

if command -v ffprobe >/dev/null 2>&1; then

duration=$(ffprobe -v quiet -show\_entries format=duration -of csv="p=0" "$win\_output\_file" 2>/dev/null)

if [[ -n "$duration" ]]; then

# Convert seconds to hours:minutes:seconds

hours=$((${duration%.\*} / 3600))

minutes=$(((${duration%.\*} % 3600) / 60))

seconds=$((${duration%.\*} % 60))

echo "Duration: ${hours}h ${minutes}m ${seconds}s"

fi

fi

fi

else

echo

echo "❌ ERROR: Concatenation failed!"

echo "Check FFmpeg output above for specific error details."

# Do not exit immediately, try to clean up

fi

# Clean up temporary files and directory

echo

echo "Step 5: Cleaning up temporary files..."

rm -rf "$TEMP\_DIR" # This will remove both $TEMP\_DIR and $CLEAN\_DIR inside it

rm -f "$TEMP\_LIST"

rm -f "$win\_temp\_list" # Remove the Windows-formatted list as well

echo "Temporary files cleaned up."

# Check if concatenation succeeded before exiting with success/failure code

if [[ -f "$OUTPUT\_FILE" && $? -eq 0 ]]; then

exit 0 # Success

else

exit 1 # Failure

fi