

Part 6B - Print-Ready Images, Page Numbers, and Contact Sheets Using ChatGPT Scripts

Yahya Nazer

2026.06.18

Table of contents

Part 6B - Print-Ready Images, Page Numbers, and Contact Sheets Using ChatGPT Scripts	2
Purpose of This Part	2
Learning Objectives	2
Course Coding Standard	3
Standard Folder Structure	3
Required Package	3
Chapter 1 - Understanding Print Size, Pixels, and DPI	4
Objective	4
Important Terms	4
Print Size	4
DPI	4
Pixel Size	4
ChatGPT Script	4
Exercise	5
What You Learned	5
Chapter 2 - Create a 5 x 7 Print Image	5
Objective	5
Python File	5
ChatGPT Script	5
ChatGPT Generated Python Code	6
Improve the Script	8
Chapter 3 - Batch Create 5 x 7 Print Images	8
Objective	8
Python File	9
ChatGPT Script	9
ChatGPT Generated Python Code	9
Improve the Script	11
Chapter 4 - Adding Page Numbers to Images	11
Objective	11
Python File	11
ChatGPT Script	11
ChatGPT Generated Python Code	12

Improve the Script	13
Chapter 5 - Creating a Contact Sheet	13
Objective	13
Python File	14
ChatGPT Script	14
ChatGPT Generated Python Code	14
Improve the Script	16
Chapter 6 - Part 6B Review	16
What You Built	16
Skills Checklist	16
Summary	16
Looking Ahead	17

Part 6B - Print-Ready Images, Page Numbers, and Contact Sheets Using ChatGPT Scripts

Purpose of This Part

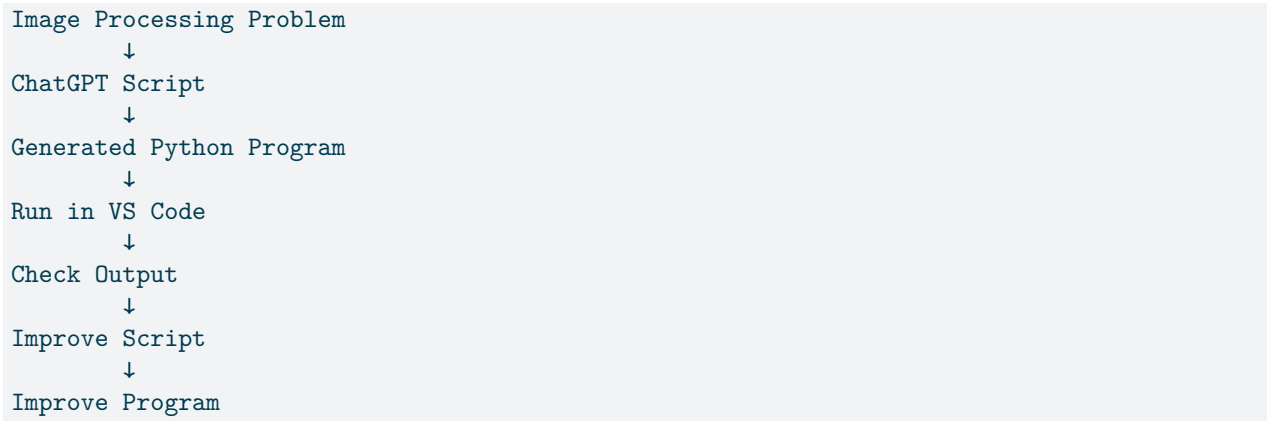
In Part 6A, you learned how to read image information, scan image folders, resize images, and convert image formats.

In Part 6B, you will use ChatGPT scripts to generate Python programs for print-ready image preparation.

You will learn how to create programs that can:

- Create fixed-size image canvases
- Preserve original image aspect ratio
- Prepare 5 x 7 print images
- Save images at 300 DPI
- Add page numbers
- Create simple contact sheets
- Save processed images to **C-Results**

The workflow remains:



Learning Objectives

After completing Part 6B, you will be able to:

- Explain canvas size for printing
 - Explain DPI in simple language
 - Create 5 x 7 images at 300 DPI
 - Preserve image aspect ratio
 - Add borders or padding
 - Add page numbers
 - Create contact sheets
 - Save print-ready images
 - Write ChatGPT scripts for image layout programs
 - Improve scripts for real photo workflows
-

Course Coding Standard

Every Python program should include:

- Program Name
- Purpose
- User ChatGPT Script
- Expected Output
- Version
- Clearly labeled steps
- Beginner-friendly comments

Example:

```
# =====  
# STEP 1 - Import Libraries  
# =====
```

Standard Folder Structure

Use this folder structure:

```
ChatGPT-Python-Course  
  A-Data  
    Images  
  B-Engine  
  C-Results  
  D-Documentation
```

Required Package

Install Pillow:

```
python -m pip install pillow
```

On some Mac computers:

```
python3 -m pip install pillow
```

Chapter 1 - Understanding Print Size, Pixels, and DPI

Objective

Understand how image size, print size, and DPI work together.

Important Terms

Print Size

Print size is the physical size on paper.

Example:

```
5 x 7 inches
8 x 10 inches
8.5 x 11 inches
```

DPI

DPI means dots per inch.

For printing, 300 DPI is common.

Pixel Size

To create a 5 x 7 inch image at 300 DPI:

```
Width  = 5 x 300 = 1500 pixels
Height = 7 x 300 = 2100 pixels
```

For landscape orientation:

```
Width  = 7 x 300 = 2100 pixels
Height = 5 x 300 = 1500 pixels
```

ChatGPT Script

Copy this into ChatGPT:

```
Explain image print size, pixels, and DPI for a beginner.
```

```
Include:
```

1. What DPI means.
2. Why 300 DPI is common for printing.
3. How to calculate pixels for 5 x 7 inches.
4. Difference between portrait and landscape.
5. Why aspect ratio matters.
6. What happens if an image is stretched.

```
Use simple language and examples.
```

Exercise

Ask ChatGPT:

```
Calculate pixel sizes at 300 DPI for:
```

1. 4 x 6
2. 5 x 7
3. 8 x 10
4. 8.5 x 11

```
Show portrait and landscape versions.
```

What You Learned

You learned how print size and pixel size are connected.

Chapter 2 - Create a 5 x 7 Print Image

Objective

Create a Python program that takes one image and places it on a 5 x 7 inch canvas at 300 DPI.

The program should:

- Preserve original aspect ratio
 - Avoid stretching
 - Add white background
 - Save output to C-Results
-

Python File

Create:

```
B-Engine/Create-5x7-Print-Image.py
```

ChatGPT Script

Copy this into ChatGPT:

```
Write a Python program.
```

```
Project folder structure:
```

```
Top Folder
```

```
+-- A-Data  
    +- Images
```

```
+-- B-Engine
```

```
+-- C-Results
```

```
+-- D-Documentation
```

```
Program Name:
```

```
Create-5x7-Print-Image.py
```

Program purpose:

Create a 5 x 7 inch print-ready image at 300 DPI from A-Data/Images/sample.jpg.

Requirements:

1. Use pathlib.
2. Use Pillow.
3. Read A-Data/Images/sample.jpg.
4. Create a white canvas sized 1500 x 2100 pixels.
5. Preserve the original image aspect ratio.
6. Fit the image inside the canvas without cropping.
7. Center the image on the canvas.
8. Save output to C-Results/sample-5x7-print.jpg.
9. Save at 300 DPI.
10. Do not modify the original image.
11. Print original size and output size.
12. If the input file is missing, show a friendly error message.

Coding Standards:

1. Add a detailed program header.
2. Include Program Name, Purpose, User ChatGPT Script, Expected Output, and Version.
3. Divide code into clearly labeled steps.
4. Add comments explaining every important line.
5. Make code beginner friendly.

Expected output:

A 5 x 7 print-ready image is saved to C-Results.

ChatGPT Generated Python Code

```
# =====  
# Program Name:  
# Create-5x7-Print-Image.py  
#  
# Purpose:  
# Create a 5 x 7 inch print-ready image at 300 DPI.  
#  
# User ChatGPT Script:  
# Read A-Data/Images/sample.jpg, create a 1500 x 2100 white  
# canvas, fit image without cropping, center it, and save 300 DPI.  
#  
# Expected Output:  
# A 5 x 7 print-ready image is saved to C-Results.  
#  
# Version:  
# 1.0  
# =====  
  
# =====  
# STEP 1 - Import Libraries  
# =====
```

```

from pathlib import Path
from PIL import Image

# =====
# STEP 2 - Define Files and Settings
# =====

input_file = Path("A-Data") / "Images" / "sample.jpg"
results_folder = Path("C-Results")
results_folder.mkdir(exist_ok=True)

output_file = results_folder / "sample-5x7-print.jpg"

canvas_width = 1500
canvas_height = 2100
dpi_value = 300

# =====
# STEP 3 - Check Input File
# =====

if not input_file.exists():
    print("ERROR: Input image was not found.")
    print("Expected file:", input_file)
else:
    # =====
    # STEP 4 - Open Image
    # =====

    image = Image.open(input_file).convert("RGB")

    original_width = image.width
    original_height = image.height

    # =====
    # STEP 5 - Calculate Resize Ratio
    # =====

    width_ratio = canvas_width / original_width
    height_ratio = canvas_height / original_height
    scale_ratio = min(width_ratio, height_ratio)

    new_width = int(original_width * scale_ratio)
    new_height = int(original_height * scale_ratio)

    # =====
    # STEP 6 - Resize Image
    # =====

    resized_image = image.resize((new_width, new_height))

    # =====
    # STEP 7 - Create White Canvas

```

```

# =====
canvas = Image.new("RGB", (canvas_width, canvas_height), "white")

# =====
# STEP 8 - Center Image on Canvas
# =====

x_position = (canvas_width - new_width) // 2
y_position = (canvas_height - new_height) // 2
canvas.paste(resized_image, (x_position, y_position))

# =====
# STEP 9 - Save Print Image
# =====

canvas.save(output_file, dpi=(dpi_value, dpi_value), quality=95)

print("Original size:", original_width, "x", original_height)
print("Output size  :", canvas_width, "x", canvas_height)
print("DPI           :", dpi_value)
print("Saved to      :", output_file)

# =====
# STEP 10 - End Program
# =====

print("Program finished.")

```

Improve the Script

Ask ChatGPT:

Please improve the program.

New requirements:

1. Allow landscape or portrait selection.
2. If landscape is selected, use 2100 x 1500 pixels.
3. If portrait is selected, use 1500 x 2100 pixels.
4. Add a thin gray border around the fitted image.
5. Keep the program header and step comments.

Chapter 3 - Batch Create 5 x 7 Print Images

Objective

Create a Python program that processes all images in `A-Data/Images`.

The program should:

- Create 5 x 7 print-ready images

- Preserve aspect ratio
 - Add white background
 - Save output to a timestamped folder in C-Results
-

Python File

Create:

```
B-Engine/Batch-Create-5x7-Print-Images.py
```

ChatGPT Script

Write a Python program.

Program Name:

```
Batch-Create-5x7-Print-Images.py
```

Program purpose:

Create 5 x 7 inch print-ready images at 300 DPI from all images in A-Data/Images.

Requirements:

1. Use pathlib.
2. Use Pillow.
3. Process images in A-Data/Images and subfolders.
4. Include extensions .jpg, .jpeg, .png, .tif, .tiff, .webp.
5. Create a 1500 x 2100 white canvas for each image.
6. Preserve aspect ratio.
7. Fit each image inside the canvas without cropping.
8. Center each image on the canvas.
9. Save output images to a timestamped folder in C-Results.
10. Keep the original filename.
11. Save output images at 300 DPI.
12. Print each processed file.
13. Count processed files and errors.
14. Do not modify original images.

Coding Standards:

1. Add a detailed program header.
 2. Divide code into clearly labeled steps.
 3. Add beginner-friendly comments.
-

ChatGPT Generated Python Code

```
# =====  
# Program Name:  
# Batch-Create-5x7-Print-Images.py  
#  
# Purpose:  
# Create 5 x 7 inch print-ready images at 300 DPI from all images.
```

```

#
# User ChatGPT Script:
# Process images in A-Data/Images, fit each on a 1500 x 2100 canvas,
# preserve aspect ratio, save at 300 DPI, and keep originals unchanged.
#
# Expected Output:
# A timestamped folder of print-ready images is saved to C-Results.
#
# Version:
# 1.0
# =====

from pathlib import Path
from datetime import datetime
from PIL import Image

input_folder = Path("A-Data") / "Images"
results_folder = Path("C-Results")
results_folder.mkdir(exist_ok=True)

timestamp = datetime.now().strftime("%Y-%m-%d--%H-%M")
output_folder = results_folder / f"Print-5x7-{timestamp}"
output_folder.mkdir(exist_ok=True)

canvas_width = 1500
canvas_height = 2100
dpi_value = 300

image_extensions = {".jpg", ".jpeg", ".png", ".tif", ".tiff", ".webp"}

def fit_image_on_canvas(input_file, output_file):
    """Fit one image on a white 5 x 7 canvas and save it."""
    image = Image.open(input_file).convert("RGB")
    width_ratio = canvas_width / image.width
    height_ratio = canvas_height / image.height
    scale_ratio = min(width_ratio, height_ratio)
    new_width = int(image.width * scale_ratio)
    new_height = int(image.height * scale_ratio)
    resized_image = image.resize((new_width, new_height))
    canvas = Image.new("RGB", (canvas_width, canvas_height), "white")
    x_position = (canvas_width - new_width) // 2
    y_position = (canvas_height - new_height) // 2
    canvas.paste(resized_image, (x_position, y_position))
    canvas.save(output_file, dpi=(dpi_value, dpi_value), quality=95)

processed_count = 0
error_count = 0

for input_file in input_folder.rglob("*"):
    if not input_file.is_file():
        continue
    if input_file.suffix.lower() not in image_extensions:
        continue

```

```
try:
    output_file = output_folder / input_file.name
    fit_image_on_canvas(input_file, output_file)
    print("Processed:", input_file.name)
    processed_count += 1
except Exception as e:
    print("ERROR processing:", input_file, e)
    error_count += 1

print()
print("Images processed:", processed_count)
print("Errors          :", error_count)
print("Output folder   :", output_folder)
print("Program finished.")
```

Improve the Script

Ask ChatGPT:

```
Please improve the batch print image program.
```

```
New requirements:
```

1. Add a GUI folder selector.
2. Let user choose portrait or landscape.
3. Show selected folder in GUI.
4. Add a log window.
5. Save log file to output folder.
6. Keep the program header and step comments.

Chapter 4 - Adding Page Numbers to Images

Objective

Create a Python program that adds a gray page number to the bottom-right corner of each image.

Python File

Create:

```
B-Engine/Add-Page-Numbers.py
```

ChatGPT Script

```
Write a Python program.
```

```
Program Name:
```

```
Add-Page-Numbers.py
```

Program purpose:
Add page numbers to images and save copies.

- Requirements:
1. Use pathlib.
 2. Use Pillow.
 3. Read images from A-Data/Images.
 4. Process extensions .jpg, .jpeg, .png.
 5. Sort images by filename.
 6. Add text "Page NN" to bottom-right corner.
 7. Use gray text.
 8. Save images to C-Results/Page-Numbered-Images.
 9. Do not modify originals.
 10. Add program header and step comments.

ChatGPT Generated Python Code

```
# =====
# Program Name:
# Add-Page-Numbers.py
#
# Purpose:
# Add page numbers to images and save copies.
#
# User ChatGPT Script:
# Read images from A-Data/Images, sort by filename, add Page NN
# to the bottom-right corner, and save copies to C-Results.
#
# Expected Output:
# Images with page numbers are saved to C-Results.
#
# Version:
# 1.0
# =====

from pathlib import Path
from PIL import Image, ImageDraw, ImageFont

input_folder = Path("A-Data") / "Images"
output_folder = Path("C-Results") / "Page-Numbered-Images"
output_folder.mkdir(parents=True, exist_ok=True)

image_extensions = {".jpg", ".jpeg", ".png"}
text_color = (120, 120, 120)

image_files = []
for file_path in input_folder.iterdir():
    if file_path.is_file() and file_path.suffix.lower() in image_extensions:
        image_files.append(file_path)

image_files = sorted(image_files)
```

```

page_number = 1

for input_file in image_files:
    image = Image.open(input_file).convert("RGB")
    draw = ImageDraw.Draw(image)
    font_size = max(24, image.width // 40)
    try:
        font = ImageFont.truetype("Arial.ttf", font_size)
    except:
        font = ImageFont.load_default()

    page_text = f"Page {page_number:02d}"
    text_box = draw.textbbox((0, 0), page_text, font=font)
    text_width = text_box[2] - text_box[0]
    text_height = text_box[3] - text_box[1]
    margin = 30
    x_position = image.width - text_width - margin
    y_position = image.height - text_height - margin
    draw.text((x_position, y_position), page_text, fill=text_color, font=font)
    output_file = output_folder / input_file.name
    image.save(output_file, quality=95)
    print("Processed:", input_file.name, "Page", page_number)
    page_number += 1

print("Images processed:", len(image_files))
print("Output folder  :", output_folder)
print("Program finished.")

```

Improve the Script

Ask ChatGPT:

Please improve the page number program.

New requirements:

1. Add page numbers only after images are converted to 5 x 7 print format.
2. Let the user choose starting page number.
3. Add source image name in gray under the page number.
4. Save output in timestamped folder.
5. Keep the program header and step comments.

Chapter 5 - Creating a Contact Sheet

Objective

Create a Python program that makes a contact sheet from images.

A contact sheet is a page containing many small images with file names under them.

Python File

Create:

```
B-Engine/Create-Contact-Sheet.py
```

ChatGPT Script

Write a Python program.

Program Name:

```
Create-Contact-Sheet.py
```

Program purpose:

Create a contact sheet from images in A-Data/Images.

Requirements:

1. Use pathlib.
 2. Use Pillow.
 3. Read images from A-Data/Images.
 4. Process extensions .jpg, .jpeg, .png.
 5. Create thumbnails.
 6. Arrange thumbnails in a grid with 4 columns.
 7. Add filename under each thumbnail.
 8. Use white background.
 9. Save contact sheet to C-Results/Contact-Sheet.jpg.
 10. Do not modify original images.
 11. Print how many images were added.
 12. Add header and step comments.
-

ChatGPT Generated Python Code

```
# =====  
# Program Name:  
# Create-Contact-Sheet.py  
#  
# Purpose:  
# Create a contact sheet from images in A-Data/Images.  
#  
# User ChatGPT Script:  
# Read images, create thumbnails, arrange in a 4-column grid,  
# add filenames, and save a contact sheet to C-Results.  
#  
# Expected Output:  
# A contact sheet image is saved to C-Results.  
#  
# Version:  
# 1.0  
# =====  
  
from pathlib import Path
```

```

from PIL import Image, ImageDraw, ImageFont

input_folder = Path("A-Data") / "Images"
results_folder = Path("C-Results")
results_folder.mkdir(exist_ok=True)
output_file = results_folder / "Contact-Sheet.jpg"

image_extensions = {".jpg", ".jpeg", ".png"}
columns = 4
thumbnail_width = 300
thumbnail_height = 220
label_height = 40
padding = 30

image_files = []
for file_path in input_folder.iterdir():
    if file_path.is_file() and file_path.suffix.lower() in image_extensions:
        image_files.append(file_path)

image_files = sorted(image_files)
rows = (len(image_files) + columns - 1) // columns
cell_width = thumbnail_width + padding
cell_height = thumbnail_height + label_height + padding
sheet_width = columns * cell_width + padding
sheet_height = rows * cell_height + padding

contact_sheet = Image.new("RGB", (sheet_width, sheet_height), "white")
draw = ImageDraw.Draw(contact_sheet)

try:
    font = ImageFont.truetype("Arial.ttf", 18)
except:
    font = ImageFont.load_default()

for index, image_file in enumerate(image_files):
    row = index // columns
    column = index % columns
    x = padding + column * cell_width
    y = padding + row * cell_height
    image = Image.open(image_file).convert("RGB")
    image.thumbnail((thumbnail_width, thumbnail_height))
    contact_sheet.paste(image, (x, y))
    label_y = y + thumbnail_height + 8
    draw.text((x, label_y), image_file.name, fill=(80, 80, 80), font=font)

contact_sheet.save(output_file, quality=95)
print("Images added:", len(image_files))
print("Contact sheet saved to:", output_file)
print("Program finished.")

```

Improve the Script

Ask ChatGPT:

```
Please improve the contact sheet program.
```

```
New requirements:
```

1. Let the user choose number of columns.
2. Add page title at the top.
3. Add image number under each image.
4. Save as PDF as well as JPG.
5. Keep the program header and step comments.

Chapter 6 - Part 6B Review

What You Built

In Part 6B, you created ChatGPT scripts and Python programs for:

- Understanding print size and DPI
- Creating one 5 x 7 print-ready image
- Batch creating 5 x 7 print images
- Adding page numbers to images
- Creating contact sheets

Skills Checklist

By the end of Part 6B, you should be able to:

- Calculate print image pixel size
- Explain 300 DPI
- Preserve aspect ratio
- Create a fixed-size canvas
- Fit an image on a canvas without stretching
- Save print-ready images
- Add page numbers to images
- Create contact sheets
- Save outputs to C-Results
- Improve image scripts with ChatGPT

Summary

In Part 6B, you learned how to use ChatGPT scripts to generate Python programs for print-ready images and contact sheets.

A safe image preparation workflow should follow this pattern:

```
Read Original Image
```

```
↓
```

```
Create New Canvas
```

```
↓
```

```
Fit Image Without Distortion
```

↓
Add Text if Needed
↓
Save Copy to C-Results
↓
Keep Original Unchanged

Looking Ahead

In Part 6C, you will learn how to build photo folder scanners, photo inventory reports, duplicate detection tools, and photo organization programs.