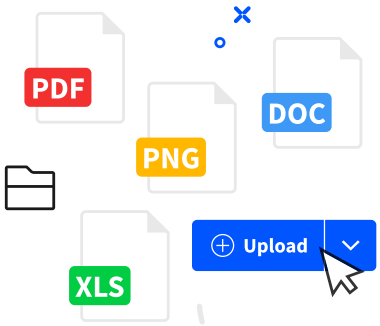
# Welcome to Smallpdf

Ready to take document management to the next level?

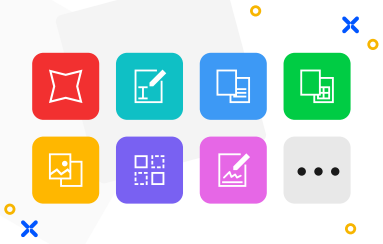


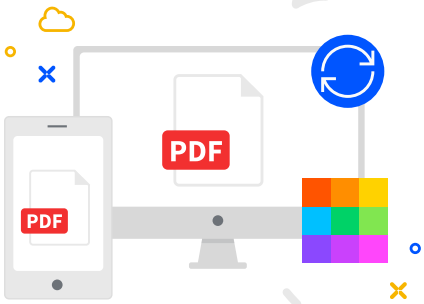

## 1 Digital Documents—All In One Place

With the new Smallpdf experience, you can freely upload, organize, and share digital documents. When you enable the 'Storage' option, we'll also store all processed files here.

## 2 Enhance Documents in One Click

When you right-click on a file, we'll present you with an array of options to convert, compress, or modify it.






## 3 Access Files Anytime, Anywhere

You can access files stored on Smallpdf from your computer, phone, or tablet. We'll also sync files from the Smallpdf Mobile App to our online portal

## 4 Collaborate With Others

Forget mundane administrative tasks. With Smallpdf, you can request e-signatures, send large files, or even enable the Smallpdf G Suite App for your entire organization.

**PDF and Image Orientation Correction Assignment**

**Background**

When scanning documents or working with digital images, they often have orientation issues (skewed, rotated, or containing unnecessary whitespace). Your task is to create a Python utility that automatically corrects these issues for both images and PDFs.

**Assignment**

Create a Python class RotationHandler that can:

1. Detect and fix image/PDF orientation issues
2. Remove skew (deskew images)
3. Crop unnecessary whitespace
4. Resize images to reasonable dimensions
5. Enhance image quality for better readability
6. Process both single images and multi-page PDFs efficiently

**Requirements**

- Your solution should handle both PDF files and common image formats (JPG, PNG)
- It should correct rotation based on text orientation detection
- It should handle EXIF rotation data in photos
- It should optimize memory usage when processing large PDFs
- Include proper error handling and logging
- Support verification of orientation correction

**Starter Code**

```
# Standard library imports - complete as needed

import io

import os

import sys

from typing import Generator, Optional
```

# Required third-party libraries

# You'll need: cv2, img2pdf, numpy, pytesseract, deskew, pdf2image, PIL

```
class RotationHandler:
    """Handles rotation correction for images and PDFs.

    Your class documentation here...
    """

    # Add appropriate constants here

    def __init__(self, verify_rotation: bool = False):
        """Initialize with configuration options."""
        # Your code here
        pass
    # Implement the following methods:
    # - _fix_exif_rotation
    # - _crop_whitespace
    # - _resize_image
    # - _enhance_image
    # - _rotate_image
    # - _deskew_image
    # - _detect_orientation
    # - _is_blank_image
    # - _process_image
    # - handle_image_rotation
    # - _pdf_to_images_generator
    # - handle_pdf_rotation
```

**Hints**

1. Use Tesseract OCR for text orientation detection (pytesseract.image_to_osd)
2. The deskew library can help determine skew angles

3. Process PDFs page-by-page to minimize memory usage
4. Use pdf2image for PDF to image conversion and img2pdf for converting back to PDF
5. Consider using a combination of contrast enhancement, sharpening, and noise reduction for image enhancement
6. Handle EXIF orientation data to fix photos taken with smartphones or digital cameras
7. Use OpenCV for advanced image processing operations

**Instructions for Testing**

1. Set up a Python environment with the required dependencies
2. Create a test folder with sample PDFs and images with orientation issues
3. Test your implementation on these files and verify the results

**Deliverables**

- Complete Python module with the RotationHandler class implementation
- Brief documentation explaining your approach
- Sample execution code showing how to use your implementation

**Evaluation Criteria**

- Correctness: Does the code properly detect and fix orientation issues?
- Robustness: Does it handle different types of files and edge cases?
- Code quality: Is the code well-structured, documented, and maintainable?
- Memory efficiency: Does it handle large PDFs without excessive memory usage?
- Error handling: Does it gracefully handle potential errors?

**Resources**

- PyTesseract documentation: https://github.com/madmaze/pytesseract
- Deskew documentation: https://github.com/sbrunner/deskew
- pdf2image documentation: https://github.com/Belval/pdf2image
- OpenCV documentation: https://docs.opencv.org/

Good luck!