**Final Lab Test: Image Processing**

**General Instructions**

1. **Duration**: The test will last **2 hours**. Late submissions will not be accepted.
2. **Generative AI Policy**:
   * You are allowed to use generative AI tools (e.g., ChatGPT, GitHub Copilot) to seek assistance or clarification for **specific coding or conceptual issues**.
   * **You may not copy/paste the task directly into the AI tool**.
   * Use AI to help you implement ideas or debug code, not to generate the full answer.
3. **Questions Policy**:
   * **No questions** will be answered by the instructor or invigilator during the test. Please read the instructions carefully.
   * Resolve any ambiguities using your judgment and reasoning.
4. **Submission Requirements**:
   * Submit a zip file containing:
     + Python scripts: task1.py and task2.py.
     + Output images: task1\_output.png and task2\_output.png.
     + A brief report (report.txt) summarizing your approach and challenges.
   * Ensure your code is well-documented and follows the required structure.
5. **Evaluation**: Each task carries **5% of the final grade**. The grading rubric is provided at the end of the document.

**Task 1: Dynamic Color Modification**

**Objective**

You will dynamically modify the color properties of an image (image1.png) to create a visually distinct version based on user-defined parameters.

**Steps**

1. **Input Parameters**:
   * Allow the user to input:
     + Color channel intensity adjustment (e.g., R:+50, G:-30, B:0).
     + Option to invert colors (Yes/No).
     + Brightness scaling factor (e.g., 1.0 to 2.0).
2. **Required Processing**:
   * Adjust the red, green, and blue channels based on user input.
   * If the user selects **inversion**, invert all colors in the image.
   * Apply brightness scaling after color adjustments.
3. **Output**:
   * Save the modified image as task1\_output.png.
   * Write a **function report()** in the code that:
     + Prints the user inputs (channel adjustments, inversion option, brightness scale).
     + Lists the steps performed in order.
4. **Example Execution**:  
   python task1.py image1.png

The script should handle user inputs interactively during execution.

**Task 2: Custom Region Enhancement**

**Objective**

You will enhance a specific region of an image (image2.png) as defined by the user, applying multiple transformations to highlight the region.

**Steps**

1. **Input Parameters**:
   * Allow the user to input:
     + Coordinates of a rectangular region (top-left x, y; width; height).
     + Desired enhancement (Blur, or Sharpen).
     + Transparency level for blending the processed region back into the original image (e.g., 0.5 for semi-transparent).
2. **Required Processing**:
   * Extract the user-defined region from the image.
   * Apply the selected enhancement (e.g., Gaussian blur, or sharpening kernel).
   * Blend the processed region with the original image using the user-defined transparency level.
3. **Output**:
   * Save the processed image as task2\_output.png.
   * Write a **function report()** in the code that:
     + Prints the user inputs (region coordinates, enhancement type, transparency level).
     + Lists the steps performed in order.
4. **Example Execution**:  
     
   python task2.py image2.png

The script should handle user inputs interactively during execution.

**Submission Guidelines**

1. Your zip file should include:
   * **Code files**: task1.py and task2.py.
   * **Output files**: task1\_output.png and task2\_output.png.
   * **Report**: A short text or PDF file (report.txt) with:
     + A summary of the steps followed in each task.
     + Challenges faced and how you used generative AI for assistance.
2. Ensure your code is modular, with functions for each processing step.
3. All files must be named exactly as specified.

**Grading Rubric**

| **Criteria** | **Task 1 (5%)** | **Task 2 (5%)** |
| --- | --- | --- |
| **Functionality** | 2% | 2% |
| **Dynamic User Input Handling** | 1% | 1% |
| **Accuracy of Results** | 1% | 1% |
| **Code Quality & Documentation** | 1% | 1% |