**1. Стиль и Организация Кода**

1. **Именование**:
   * Используйте понятные и самодокументирующиеся имена переменных, функций и классов.
   * Придерживайтесь camelCase для переменных и методов (пример: generateImage()) и PascalCase для классов (пример: ImageGenerator).
   * Константы и enum-значения оформляйте в UPPER\_CASE (пример: MAX\_BUFFER\_SIZE).
2. **Структура файлов**:
   * Каждый файл должен содержать одну основную сущность (класс или модуль), логически связанную с её функционалом.
   * Группируйте функции, относящиеся к одному модулю, чтобы облегчить понимание кода.
3. **Комментарии**:
   * Комментируйте сложные блоки кода и алгоритмы.
   * Избегайте избыточных комментариев, дублирующих очевидные операции.

**2. Форматирование**

1. **Отступы и пробелы**:
   * Используйте одинаковый стиль отступов (чаще всего 4 пробела).
   * Старайтесь не допускать длинных строк (рекомендуется максимум 80–120 символов).
2. **Пустые строки**:
   * Разделяйте логические блоки пустыми строками для лучшей читаемости.
3. **Фигурные скобки**:
   * Располагайте их единообразно; например, открывающая скобка на той же строке, что и оператор (стиль K&R).

**3. Управление Версиями и Структура Проекта**

1. **Использование Git**:
   * Делайте частые коммиты с осмысленными сообщениями: описывайте, что именно было сделано.
   * Оформляйте pull request и проводите код-ревью, чтобы улучшить качество и обсудить решения.
2. **Организация репозитория**:
   * Разделяйте код на директории: src/, include/, tests/ и т.д.
   * Документацию и дополнительную информацию храните в docs/.

**4. Обработка Исключений и Логирование**

1. **Обработка ошибок**:
   * Учитывайте все возможные исключения (например, ошибки ввода-вывода).
   * Предоставляйте понятные сообщения об ошибках, чтобы облегчить отладку.
2. **Логирование**:
   * Используйте логи для важных действий: загрузка данных, начало/конец длительных операций, критические ошибки.
   * Не перегружайте логи избыточной информацией.

**5. Безопасность и Тестирование**

1. **Безопасность**:
   * Не храните пароли и ключи в открытом виде в коде.
   * Проверяйте входные данные и параметры на валидность.
2. **Тестирование**:
   * Пишите модульные тесты (Unit Tests) для ключевых функций и классов.
   * Используйте автоматическую систему сборки (например, CMake, GitHub Actions) для прогонки тестов при каждом коммите.