# Reproducible Research Workflows

**Note on data.** This assignment uses a **synthetic** voter-turnout dataset provided by `poliscitools` (via `example_data`). The goal is to practice reproducibility tooling (dependency management, logging, packaging outputs, and replication checks)—not to draw substantive inferences about real voters.

## Conceptual Questions

Please write three to ten sentence explanations for each of the following questions. **You are only required to answer ONE of the two questions below.**

1. Explain what problem `renv` solves in reproducible research. In your answer, describe what information is stored in `renv.lock`, what `renv::restore()` does, and why sharing code without dependency versions can fail replication even when the analysis is "correct."
2. Explain why logging (e.g., `logger`) is part of professional, reproducible analysis. Give two concrete examples of what you would log in a pipeline (inputs, parameters, random seeds, file paths, model summaries, warnings), and explain how logs help diagnose non-reproducible results.

## Applied Exercises

Use the code in the week's code tutorial and the lecture slides to answer the following questions.

3. **Reproducible project setup: dependencies, directories, and end-to-end run.** Using the provided reproducibility script:
   - Create a clean project folder and initialize `renv` (or restore from an existing `renv.lock` if provided).
   - Run the full pipeline by sourcing the script and running `run_analysis()`.
   - Verify that the following artifacts are produced:
     - `data/raw/voter_data.csv` and `data/processed/cleaned_voter_data.csv`,
     - figures in `outputs/figures/`,
     - tables in `outputs/tables/` (including the bootstrap output),
     - `outputs/session_info.txt` and `analysis_log.txt`.
   - Briefly describe (3–6 sentences) what each folder is for (`data/raw`, `data/processed`, `outputs/figures`, `outputs/tables`) and why the separation matters for reproducibility.
4. **Extend outputs: add one additional plot and one additional table.** Modify the script to generate:
   - **One additional plot** saved to `outputs/figures/`. Choose one:
     (a) a plot of bootstrap coefficient distributions (from `boot_coef`),
     (b) a turnout plot broken down by two dimensions (e.g., party × age_group),

    (c) a plot showing model residuals or predicted vs. observed turnout.

  - **One additional table** saved to `outputs/tables/`. Choose one:
    (a) a turnout summary table by party and age group,
    (b) a table of regression coefficients with confidence intervals (or bootstrap intervals),
    (c) a missingness summary table for the cleaned dataset.
  - In 3–6 sentences, explain how these additions improve the interpretability and auditability of the analysis.

5. **Reproducibility checks: logging + repeated runs.** Use the reproducibility testing ideas in the script:
   - Run the analysis multiple times (at least 3) using the provided reproducibility testing function (or an equivalent workflow you implement).
   - Report whether results are identical across runs. If they are not identical, identify **one** source of non-determinism and fix it (e.g., missing `set.seed()`, randomness in resampling, unstable ordering).
   - Add at least **two** additional log messages (`log_info`) that record key intermediate facts (e.g., row counts before/after cleaning, number of bootstrap iterations completed, output file paths written).
   - In 5–8 sentences, explain what you learned from the reproducibility check and what would still threaten reproducibility across different machines.
6. **Challenge Question (Optional — if you finish early):** Containerize and/or automate reproducibility. Choose **ONE** of the following:
   (a) **Docker replication.** Use the provided `create_dockerfile()` logic (or write your own Dockerfile) to run the full analysis in a container. Provide evidence it worked (e.g., outputs created in `outputs/` and a short excerpt of container logs). In 4–8 sentences, explain what Docker adds beyond `renv`.
   (b) **GitHub Actions replication.** Set up a minimal GitHub Actions workflow that (i) installs R, (ii) restores `renv`, and (iii) runs `run_analysis()`. Provide the workflow YAML file and a screenshot (or log excerpt) showing a successful run.

**Submission:** Push your code changes to GitHub and submit a link to your repository. Your repository should include `renv.lock`, your modified script(s), and the updated outputs you created (figures/tables). Do not commit large, unnecessary intermediate files.