MTH208 Mid-Semester Examination

Instructions

1. **Seating:** Sit according to the seating plan. Please arrive at the lab by **5:40 PM** to allow enough time to locate your assigned seat.
2. **Template & naming**
   - Download the template file **`ms_template.R`** and the data files from the helloIITK exam page. These will be available five minutes before the exam begins.
   - Rename it to **`ms_<ROLL>.R`**, where `<ROLL>` is your roll number (e.g., `ms_230123.R`).
   - Work only in this renamed file.
3. **What to edit**
   - Follow the instructions for each question **exactly**.
   - **Only** edit the lines marked with dashes `-------` or comments `# your code here`.
   - **Do not** change function or object names required by the grader: `cityDejaVu`, `hsl_convert`, `students_clean` (and any other names specified in questions).
   - **Do not** add `rm(list = ls())`, `setwd()`, or absolute paths.
4. **Code style**
   - Comment your code clearly; poor commenting/spacing may lose points.
5. **Libraries**
   - Use only **`dplyr`** and **`imager`**. Base R helpers (e.g., `gsub`, `sub`, `trimws`, `toupper`) are allowed **inside** your `mutate()` calls.
   - Do **not** use other packages (`stringr`, `tidyr`, `janitor`, `data.table`, etc.).
6. **I/O and paths**
   - Your script must run in a fresh R session with the working directory set to the folder containing your files.
   - Use the provided relative paths (e.g., `data/...`) exactly as stated in the questions.
7. **Submission**

- Submit **only one file**: your renamed script `ms_<ROLL>.R`.
- Upload it on the **helloIITK** Mid Sem Exam page. Submissions by email or any other method will **not** be accepted.

8. **Grading run**
   - Your script **must run end-to-end** on the grader's machine with:

     `source("ms_<ROLL>.R")`

     without manual edits, assuming the provided data and template structure.
9. **Evaluation data: Your code will be evaluated on different data files** (same schema/column names, different values). Please avoid hard-coding values or assumptions beyond what's specified; rely on the given column names and rules, and use relative paths only.

Q1. Simulation (3 points)

Story. It's 2036 and you've moved to Hyderabad. You have two social circles there:

- Circle A: 120 former classmates
- Circle B: 40 ex-colleagues

Assumptions (all encounters uniformly random from the locality)**:**

- Locality size: 200,000 people
- Daily encounters: 500 people (independent from day to day)
- Meeting anyone from either Circle A or Circle B ends the simulation

Task. Write an R function `cityDejaVu()` with no inputs that simulates day by day and until you first meet someone from Circle A or B and returns:

1. `p_day`, the probability that, on a single day, you meet at least one person from Circle A or B.
2. `days` until the first "hit" (each day is a Bernoulli trial with success prob `p_day`).

Notes:

- Comment your steps clearly.
- Return a list:
  `list(days = <integer number of days>, p_day = <computed probability>)`

Q2. Image HSL Histograms (3 points)

Task: Write an R function `hsl_convert(img)` that takes an `imager` image object as input and returns nothing; it plots three histograms in one row: Hue, Saturation, Lightness, each with axis labels and descriptive titles.

Color-space conversion (per pixel), with $R, G, B \in [0, 1]$:

1) $C_{\max} = \max(R, G, B), \quad C_{\min} = \min(R, G, B), \quad \Delta = C_{\max} - C_{\min}$
2) Lightness: $L = \dfrac{C_{\max} + C_{\min}}{2}$
3) Saturation S:

$$S = \begin{cases} 0, & \Delta = 0, \\ \dfrac{\Delta}{1 - |2L - 1|}, & \Delta > 0. \end{cases}$$

4) Hue H in degrees $[0, 360)$: If $\Delta = 0 \Rightarrow H = 0$. Otherwise,

$$H = 60 \times \begin{cases} \dfrac{G - B}{\Delta} \bmod 6, & C_{\max} = R, \\ \dfrac{B - R}{\Delta} + 2, & C_{\max} = G, \\ \dfrac{R - G}{\Delta} + 4, & C_{\max} = B. \end{cases}$$

Finally wrap to $[0, 360)$.

Image HSL Histograms

Notes:

- Use fixed binning: Hue — `seq(0, 360, 10)`; Saturation — `seq(0, 1, 0.05)`; Lightness — `seq(0, 1, 0.05)`. Plot counts (not densities) and set `include.lowest=TRUE` with `par(mfrow=c(1,3))`.
- Labeled axes, and clear titles ("Hue (deg)", "Saturation", "Lightness").
- Your function must not write files and must not return a value.

Q3. Data from CSV (4 points)

Input: `data/students_raw_simple.csv` (first three columns are already clean: `ID, Name, Dept`; you only need to clean `Section, Score, Attendance`).

Goal: Produce a tibble named `students_clean` with exactly these 7 columns in this order: `ID, Name, Dept, Section, Score, Attendance, Pass`.

Cleaning rules (use dplyr verbs only; base helpers allowed inside `mutate()`):

1. Do not modify `ID`, `Name`, or `Dept`. They are already standardized.
2. Section normalization: - Keep the first alphabetic character, uppercase it (`A`/`B`/`C`).
3. Numeric parsing: - Score: keep only the leading numeric part (supports decimal comma or dot), convert comma to dot, then `as.numeric()`; e.g., `68,0`→68.0, `72/100`→72, `90.0 pts`→90.0. - Attendance: remove non-numeric characters except dot/comma, convert comma to dot, then `as.numeric()`; interpret as percentage 0–100.
4. Pass flag: - `Pass = (Score >= 40) & (Attendance >= 75)` with both non-`NA`; otherwise `FALSE`.