ANSWER KEY

MATEMATISKA INSTITUTIONEN
STOCKHOLMS UNIVERSITET
Avd. Matematik
Examinator: Per Alexandersson

Tentamensskrivning i
Statistisk databehandling
MT4007
2026-02-19

— ANSWER KEY —

Part I

1. In functional programming, what is a *side effect*?

   (a) Returning a new value computed from the input arguments.

   (b) Applying a higher-order function such as `map` or `filter`.

   (c) **Any observable change outside the function, such as modifying a global variable or writing to a file.**

   (d) Passing a function as an argument to another function.

   (e) Raising an exception when the input is invalid.

   (f) Creating a local variable inside the function body.

2. What does *function composition* mean in functional programming?

   (a) Writing a function that calls itself recursively to reduce a list.

   (b) **Combining two small functions to create a new function, where the output of one becomes the input of the next.**

   (c) Using `map` to apply a function element-wise to a list.

   (d) Defining a function inside another function to create a closure.

   (e) Filtering a list and then reducing it to a single summary value.

   (f) Passing multiple arguments to a single function call at once.

3. What does `git clone` do?

   (a) It creates your own independent copy (fork) of someone else's repository on GitHub.

   (b) **It downloads a full copy of a remote repository to your local machine.**

   (c) It creates a new branch from the current `HEAD` of the repository.

   (d) It stages all modified files and prepares them for the next commit.

   (e) It uploads local changes to the remote repository.

   (f) It replays commits from one branch onto another.

4. Why is it recommended to make small, frequent commits with clear messages, rather than one large commit at the end?

   (a) Because Git compresses small commits more efficiently, saving disk space.

   (b) Because each commit creates a restore point, and smaller ones reduce the chance of merge conflicts.

   (c) **Because it makes it easier to locate, understand, and revert specific changes if something breaks.**

   (d) Larger messages have more risk of package loss in transit.

   (e) Because `git push` has a maximum payload size that large commits may exceed.

   (f) Because the commit message field has a strict 30-character limit.

5. In a few sentences, explain what an environment is and why using one improves the reproducibility of a data analysis project.

   Sample answer: An environment (e.g., a virtual environment in Python via `venv` or `conda`, or `renv` in R) is an isolated collection of software packages and their exact versions, specific to a project. Using an environment improves reproducibility because it ensures that anyone running the code uses the same package versions as the original author. Without an environment, package updates or differences between machines can cause code to produce different results or fail entirely. By recording the environment (e.g., in a `requirements.txt` or `renv.lock` file), others can recreate it exactly and run the analysis with the same dependencies.

Part II

6. Which core operation (verb) would you use to keep only the rows in a data frame where `Age > 30`?

   (a) **Filter** (b) Select (c) Group By (d) Mutate (e) Sort (f) Summarize

7. You want to compute the average salary for each department in a company. Which combination of operations is most appropriate?

   (a) Filter rows where salary is not null, then select the salary column.

   (b) **Group by department, then summarize with mean salary.**

   (c) Mutate a new column with normalized salary, then sort descending.

   (d) Pivot wider so each department becomes a column, then compute column means.

(e) Select department and salary, then drop duplicate rows.

(f) Join the table with itself on department, then count distinct rows.

8. A line chart (with time on the x-axis) is most appropriate for visualizing:

   (a) The frequency distribution of a single numeric variable.

   (b) The relationship between two unrelated categorical variables.

   (c) **How a numeric variable changes over time.**

   (d) The relative proportions of parts within a whole.

   (e) Pairwise correlations among many numeric variables at once.

   (f) The count of observations in each category of a factor.

9. In the tidy data format, why is it a problem if a single cell contains multiple values (e.g., `"180/75"` for height and weight)?

   (a) It makes the CSV file ambiguous.

   (b) It prevents the editor from doing proper syntax high-lighting.

   (c) **It violates the tidy principle that each cell should hold exactly one value, making standard operations like filtering and grouping unreliable.**

   (d) It causes Python and R to interpret the value as division.

   (e) It is only a problem if the file uses semicolons as delimiters.

   (f) It forces the column to be parsed as text, but this is easily fixed with type casting.

10. Describe, step by step, how you would compute the average height per occupation from a data frame with columns `Name`, `Age`, `Height_cm`, and `Occupation`. Use operation names (verbs) in your answer.

Sample answer: (1) **Group By** the `Occupation` column to partition the data into one group per occupation. (2) **Summarize** (aggregate) each group by computing the **mean** of `Height_cm`. This produces a new data frame with one row per occupation and the corresponding average height. Optionally, you could first **Filter** out rows with missing `Height_cm` values, and afterwards **Sort** the result by average height for easier reading.

Part III

11. Describe two different strategies for handling missing values in a dataset. For each, give one situation where that strategy would be appropriate.

> **Sample answer: Strategy 1 — Deletion**: Remove rows (or columns) that contain missing values. This is appropriate when the proportion of missing data is small and the missingness is completely random (MCAR), so removing those rows does not introduce bias into the analysis.
>
> **Strategy 2 — Imputation**: Replace missing values with a substitute, such as the mean, median, or mode of the column (or a value predicted by a model). This is appropriate when dropping rows would lose too much data, or when missing values follow a pattern that can be estimated from other variables—for example, replacing missing income with the median income for the same occupation group (grouped imputation).

12. What does *grouped imputation* mean when handling missing data?

    (a) Dropping every group that contains at least one missing value, keeping only complete groups.

    (b) Replacing missing values with the overall dataset mean, computed across all groups.

    (c) **Replacing missing values with a summary statistic (e.g., median) computed within each group or category.**

    (d) Flagging missing values as outliers and then applying the IQR rule within each group.

    (e) Joining two tables on the grouping key and using the matched rows to fill gaps.

    (f) Imputing missing values first, and then grouping the completed data for analysis.

13. When using the IQR rule, how is the **upper** bound for flagging outliers typically calculated?

    (a) $Mean + 2 \times Standard\ Deviation$

    (b) $Q1 - 1.5 \times IQR$

    (c) $Median + 1.5 \times IQR$

    (d) $Q3 + 1.5 \times IQR$

    (e) $Q3 + 3.0 \times IQR$

    (f) $Q3 + 1.5 \times Standard\ Deviation$

14. Why is it important to check row counts after performing a join operation?

    (a) Because an inner join should always produce more rows than either input table.

    (b) Because a left join guarantees the row count stays unchanged, so any difference signals an error.

    (c) **Because unexpected row increases (from duplicated keys) or decreases (from unmatched keys) can silently corrupt the analysis.**

    (d) Because SQL will raise a warning if the resulting table exceeds the row count of the left table.

    (e) Because differing row counts indicate that the column types in the two tables are incompatible.

    (f) Because pandas and dplyr require you to declare the expected row count before executing a join.

15. Which of the following best describes a "Long to Wide" transformation?

    (a) Concatenating (stacking) two data frames that share the same columns.

    (b) **Taking a column of category labels (e.g., "Medal_Type") and spreading its unique values into separate columns with corresponding values.**

    (c) Collapsing multiple columns (e.g., "Gold", "Silver", "Bronze") into a single key-value pair of columns.

    (d) Grouping by a key column and computing an aggregate statistic for each group.

    (e) Joining two data frames on a shared key and keeping all rows from both sides.

    (f) Sorting the data frame by a categorical column so that each category appears in a contiguous block.

Part IV

16. You perform an `INNER JOIN` between Table A and Table B. What happens to rows that exist in Table A but have no matching key in Table B?

    (a) They are kept, with columns from Table B filled as `NULL`.

    (b) **They are removed from the result entirely.**

    (c) They are kept only if Table A is specified as the left table.

    (d) They are duplicated to preserve the original row count of Table A.

(e) They are moved into the result but flagged with a special `_unmatched` indicator column.

(f) They are kept, and Table B columns are filled with the default value defined in the schema.

17. What does the `NOT NULL` constraint on a SQL column ensure?

    (a) That the column values must be unique across all rows.

    (b) **That every row must have a value in that column; empty entries are not allowed.**

    (c) That the column serves as a foreign key referencing another table.

    (d) That the column is included in the table's primary key.

    (e) That the column has a default value which is inserted when no value is supplied.

    (f) That the column is automatically indexed for faster query performance.

18. Write a SQL query that returns the `Name` and `City` of all persons older than 25 from a table called `Persons` with columns `Name`, `Age`, and `City`. Sort the result by `Age` in descending order.

    Sample answer:
    ```
    SELECT Name, City
    FROM Persons
    WHERE Age > 25
    ORDER BY Age DESC;
    ```

19. In a regular expression, what does the anchor `^` match when placed at the beginning of a pattern (e.g., `^Hello`)?

    (a) Any single character.

    (b) The end of the string or line.

    (c) **The start of the string or line.**

    (d) A word boundary between a word character and a non-word character.

    (e) The negation of a character class (e.g., "not a digit").

    (f) Zero or one occurrence of the next character.

20. Explain what a *foreign key* is in a relational database. Give a concrete example involving two tables (you may use table and column names of your choice).

    Sample answer: A foreign key is a column (or set of columns) in one table that references the primary key of another table. It enforces *referential integrity*: you cannot insert a value in the foreign key column that does not already exist in the referenced table, and you cannot delete a referenced row without first removing or updating the rows that point to it.

    Example: An `Orders` table has a column `CustomerID` which is a foreign key referencing the `CustomerID` primary key in a `Customers` table. This ensures every order is linked to a valid customer—you cannot create an order for a customer that does not exist in the `Customers` table.

Part V

21. Which HTTP method is used to **retrieve** a resource from a REST API without modifying it?

    (a) `POST`

    (b) `DELETE`

    (c) `PUT`

    (d) `GET`

    (e) `PATCH`

    (f) `HEAD`

22. Why should API keys or tokens **never** be hardcoded directly in your source code that is pushed to GitHub?

    (a) Because Git tracks every version of the file, so even if deleted later the key remains in the commit history.

    (b) Because hardcoded strings cause encoding errors when the repository is cloned on a different operating system.

    (c) **Because anyone who views the public repository can see and misuse the credentials.**

    (d) Because GitHub's security scanner will automatically revoke the key and block further API calls.

    (e) Because environment variables are the only string type that Python's `requests` library can read for authentication.

    (f) Because API providers require keys to be stored in a `.env` file or they refuse the connection.

23. When web scraping, what is the purpose of checking a website's `robots.txt` file?

    (a) It lists the REST API endpoints provided by the website as alternatives to scraping.

    (b) It contains the CSS selectors needed to locate data elements on the page.

    (c) **It specifies which parts of the site automated bots are allowed or disallowed from accessing.**

    (d) It declares the rate limit (requests per second) that the server enforces.

    (e) It provides a machine-readable sitemap of all pages, in JSON format.

    (f) It indicates whether the site's content is loaded statically or dynamically via JavaScript.

24. If you need to scrape data from a page where the content is rendered entirely by JavaScript, which approach is most appropriate?

    (a) Use `requests.get()` / `read_html()` and parse the returned HTML directly.

    (b) Pass a custom `User-Agent` header that identifies your script as a modern browser, which triggers the server to include the full content.

    (c) **Use a headless browser (e.g., Selenium or Playwright) that can execute JavaScript before extracting the content.**

    (d) Inspect the page source for `<noscript>` tags, which always contain the same data in plain HTML.

    (e) Set the `Accept: application/json` header so the server returns the data as JSON instead of HTML.

    (f) Increase the `timeout` parameter so the HTTP library waits for the JavaScript to finish executing.

25. Modern REST APIs typically return data in JSON format, which uses nested objects and arrays. Explain briefly why this nested structure often needs to be "normalized" or "flattened" before it can be used as a data frame, and name one function (in Python or R) that helps with this.

Sample answer: JSON data is hierarchical: a single record may contain nested objects (e.g., an `address` object inside a `user` object) or arrays (e.g., a list of items within one order). A data frame, however, is a flat, rectangular structure where each row is one observation and each column is one variable. When you load a JSON response directly, nested sub-objects become a single cell containing a dictionary or list, which cannot be filtered, sorted, or analysed with standard data-frame operations. "Normalizing" unpacks these nested structures into separate columns (and possibly separate rows for array fields), producing a flat table. In Python, `pandas.json_normalize()` does this automatically. In R, `tidyr::unnest()` or `jsonlite::fromJSON(..., flatten = TRUE)` are common alternatives.