Software Assessment - Report Cards

You're a Jr. Dev working at a company that has just been granted a contract for building a teacher's admin tool for the local high school! Different teams have been assembled to take on different parts of the app, your team will be working on the dashboard for the school principal.

One specific feature your client wanted was to be able to quickly generate and visualize report cards from .csv files containing the student courses and grades. The front-end developers in your team are already working on displaying the reports, but they don't know how to parse .csv files and would rather be working with JSON data that is already processed. This is where you come in.

Your manager needs you to build a tool that reads these .csv files, parses them, calculates the students' final grades and generates the report as a structured JSON file that can easily be consumed by the front-end.

If you notice something is not working (like the API, or any of the links in this document), please contact hello@hatchways.io.

Evaluation

This assessment will be evaluated based on the following criteria:

- Correctness: Is your solution complete and does it pass different test cases?
- Code Organization, Readability, & Maintainability: Is your code easy to read and well organized?
- Code Performance: Is your code efficient? Did you use appropriate data structures?
- Best Practices: Did you utilize good programming practices (write unit tests, avoid anti-patterns)? Did you show a good grasp of your language/framework of choice?
- Completion speed: A fast completion time comparable to the completeness of your solution. This is the least important criteria.

We use the following rubric to evaluate your submission. **Please note that if your submission does not attempt to complete all of the requirements, we will be unable to provide feedback on it.**

Inputs

You will be given four files as an input to your program. The description of each file is written below. Here is a compressed folder with a simple example input and the output.

courses.csv

This file contains the courses that a student is taking. Each course has a unique **id**, a **name**, and a **teacher**.

students.csv

This file contains all existing students in the database. Each student has a unique **id**, and a **name**.

tests.csv

This file contains all the tests for each course in the courses.csv file. The file has three columns:

- **id:** the test's unique id
- **course_id:** the course id that this test belongs to
- **weight:** how much of the student's final grade the test is worth. For example, if a test is worth 50, that means that this test is worth 50% of the final grade for this course.

The sum of all the weights of every test in a particular course should add up to 100. If that's not the case, your output should be an error message (see an example in the output section below).

marks.csv

This file contains all the marks each student has received for every test they have written.

The file has three columns:

- **test_id:** the test's id
- **student_id:** the student's id
- **mark**: The percentage grade the student received for the test (out of 100)

Notes:
Not every student is enrolled in all courses – a student is considered to be enrolled in a course if they have taken a least one test for that course

Output

Given the example input, here is the JSON file you need to generate:


```
{
  "students": [
    {
      "id": 1,
      "name": "A",
      "totalAverage": 72.03,
      "courses": [
        {
          "id": 1,
          "name": "Biology",
          "teacher": "Mr. D",
          "courseAverage": 90.1
        },
        {
          "id": 3,
          "name": "Math",
          "teacher": "Mrs. C",
          "courseAverage": 74.2
        },
        {
          "id": 2,
          "name": "History",
          "teacher": "Mrs. P",
          "courseAverage": 51.8
        }
      ]
    },
    {
      "id": 2,
      "name": "B",
      "totalAverage": 62.15,
      "courses": [
        {
          "id": 1,
          "name": "Biology",
          "teacher": "Mr. D",
          "courseAverage": 50.1
        },
        {
          "id": 3,
          "name": "Math",
```

```
          "teacher": "Mrs. C",
          "courseAverage": 74.2
        }
      ]
    },
    {
      "id": 3,
      "name": "C",
      "totalAverage": 72.03,
      "courses": [
        {
          "id": 1,
          "name": "Biology",
          "teacher": "Mr. D",
          "courseAverage": 90.1
        },
        {
          "id": 2,
          "name": "History",
          "teacher": "Mrs. P",
          "courseAverage": 51.8
        },
        {
          "id": 3,
          "name": "Math",
          "teacher": "Mrs. C",
          "courseAverage": 74.2
        }
      ]
    }
  ]
}
```

Notice that:

- The data is contained within an object with a “students” key
- The students are ordered by id (the courses don’t have to be ordered)
- The ids and grades are numbers
- Grades are rounded to two digits
- The *totalAverage* is the average of all the courses the student is enrolled in
- The student’s *courseAverage* in each course is determined by the mark they get in each test, and how much each test is worth)
- Not all data from the .csv files appears in the report.

Here is another example of inputs and output for this assessment.

For an invalid input (i.e where the course weights don’t add up to 100), your output should only contain an “error” key with a message:

```
{
  "error": "Invalid course weights"
}
```

Important: Make sure the output follows the format above (e.g, it's an object with key "students" or "error") and that it is valid JSON (you can use a validator).

Testing

An important part of development is testing. We want to see some unit tests written to ensure your solution works for different cases. Think about different ways to test the app, and the best way to get good coverage.

Usage

Since your program will be used with different inputs, it must not hardcode the filenames or paths. Instead, take them as command line arguments. Arguments will be passed in the following order:

{path-to-courses-file} {path-to-students-file} {path-to-tests-file} {path-to-marks-file} {path-to-output-file}

For example:
"python main.py courses.csv students.csv tests.csv marks.csv output.json"
"node app.js courses.csv students.csv tests.csv marks.csv output.json"

If you attempt to hardcode the paths or get them from the standard input, your application will not pass our tests.

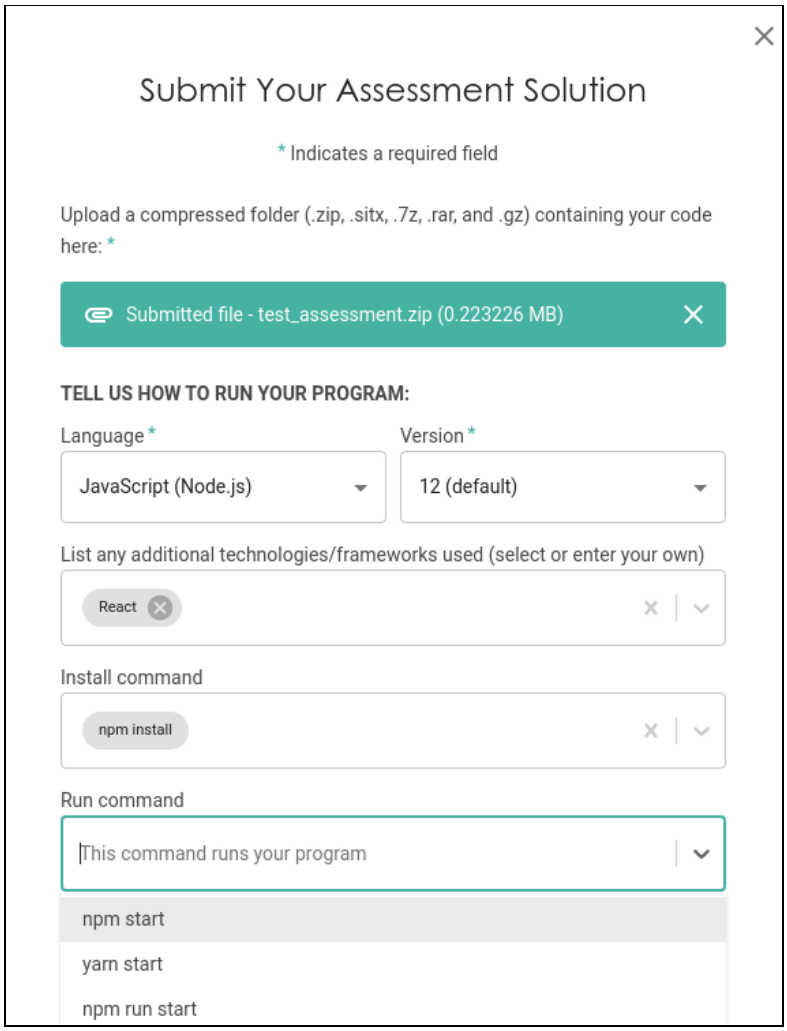
Submission Details

Please submit your code in a compressed folder on the Hatchways platform. The max submission size is 5MB.

Upon clicking the submission button, you will see a form as pictured below. We need this information to be able to test your application.

1. Choose which language and technologies you used to develop your solution. Be sure to select the appropriate version for the language you have used.
2. Provide us with the **install command** and the **run command** that you used to run your application. The run command should include only the base run command, not the file locations. For example, use just "python main.py" of "node app.js".
3. If you cannot find your commands in our suggestions, simply type your own and select “Use command”.

Please note that these commands will be used to run automated tests, so filling in every relevant field and providing accurate commands will allow you to receive feedback more quickly on your submission. If you have any notes to provide about your submission, please put them in a README, not in the submission form. Additionally, note that the install and run commands will be run from the root level of your submission, so please organize your files accordingly.



Do not submit any built folders, since the compressed folder will be too large. **Do not submit your external dependencies (like the node_modules folder), since the**

compressed folder will be too large. We will be installing your dependencies before we run your code.

If your submission is too big and you can't figure out how to compress, you are welcome to email your solution to hello@hatchways.io. Please include your name, and use the email you signed up with on the Hatchways platform. Use the subject line “Software Assessment Submission”.

Public Repositories

Do not post your solution to a public repository. We understand that you may want to share projects you have worked on, but many hours go into developing our tools so we can provide a fair skills evaluation.