Exercise Sheet #9

Deadline: 17.06.2024, 12:00h

Project A small programming project is mandatory to complete the course. Your task is to implement a client for a multiplayer version of rock paper scissors on a 2D playing field. The framework for the game is provided in this repository.

- You can work in *groups* of up to 3 people.
- You can choose whatever *strategy* you like: rule based, random movements, deep learning approaches, ... It's up to you!
- You are also free to choose the *programming language* to work in. Of course you can solve the project in `C++`, but other languages might be easier. Note, however, that skeleton clients are only provided for `Python` and `C++`.
- The *minimum requirement* for the project is to present a working, non-trivial client.
- The *deadline* to submit your project is on the 12th of July 2024. Submit your project via e-mail to
  `nevermann@itp.uni-frankfurt.de`.
  Make sure to test your client. For testing, you can use the provided test client or play against your own client. We will do a tournament (rock paper scissors world cup ☺) between the different clients on the 16th of July 2024 in the lecture. The winner will receive a small price!

Problem 1 (*Containers: Word Frequency Analysis*) (10 points)

In this problem we will use different containers in `C++` to implement a word frequency analyzer.
The program should ask the user for a text and output a list of all words in the text with respective numbers of occurrence. Follow these steps:

(a) Ask the user for input and store the input text in a `std::string`. Then split the text into words and store all words in a `std::vector<std::string>`. Consider a word as a sequence of characters separated by spaces. Ignore punctuation.

(⑤ points)

(b) Use an `std::iterator` to iterate over the vector of words. Count the number of occurrences and store them into a `std::map<std::string, int>`.

(③ points)

(c) Print the word frequencies to the console.

(② points)

Problem 2 (*Matrix Inversion using Gauss Elimination*) (10 points)

In the lecture you saw how to implement Gauss elimination with partial pivoting in `C++` (link) and used the algorithm to solve systems of linear equations. Gauss elimination can however also be used to find the inverse of a matrix $A \in \mathbb{R}^{n\times n}$, as the matrix inversion problem is equivalent to solving n linear systems of size n, each of the form

$$AA_i^{-1} = I_i, \qquad i = 1, \ldots, n,$$

where A_i^{-1} and I_i are the i-th columns of the inverse matrix of A and the identity matrix, respectively.
Implement a function

```
invertMatrix(vector<vector<double>>& A, vector<vector<double>>& Ai)
```

that takes a matrix `A` and a matrix `Ai` as inputs and inverts the matrix in-place using Gauss elimination with partial pivoting, storing the resulting inverse matrix in `Ai`. Provide at least one meaningful example.