CPU Algorithm Design

Exercise 1 **Student:** Konstantin Benz

Task 1

(1) You use std::optional<T> to represent an object of type T that may or may not be present. It is a wrapper around the type T that can be either empty or contain a value. This is useful when you want to indicate that a value might not be available without using pointers or special values (like nullptr or -1). For instance, a find_user(id) function can return std::optional<User>-std::nullopt if the user does not exist, otherwise the found User object.

(2) std::variant is a type-safe union that can hold exactly one of several types. A std::tuple is a fixed-size collection of the same types. Therefore a std::variant is more compact memory-wise than a std::tuple, because it only needs to store the size of the largest type, while a std::tuple needs to store the size of each type.

(3) std::pair and std::complex give their two components a fixed semantic meaning (first/second, real/imaginary) and always contain exactly two elements. In contrast, std::tuple and std::array are purely structural containers whose members are accessed by position and have no predefined interpretation.

(4) std::pair and std::tuple can store heterogeneous types (each element may have a different type). std::array<T, N> and std::complex<T> are homogeneous: every stored value has the same type T (and, for std::complex, there are always exactly two such values).

(5) std::complex<T> is a domain-specific numeric abstraction: beyond holding two values, it models the algebra of complex numbers and overloads arithmetic operators (‘+‘, ‘-‘, ‘*‘, ‘/‘, ‘abs‘, ‘arg‘, . . .). The other templates are generic containers and provide no mathematical behaviour on their own.

Task 2

(1) test

Here I show a very basic example of how to use the „`problem`“ environment I defined using the `\tcolorbox` package. You can define your own environments following the problem environment in the `format.tex` file.

Task 3: Your title

This is an example problem taken from **Sakurai2020**:

(a) Prove the following

(i) $\langle p'|x|\alpha\rangle = \mathrm{i}\hbar\frac{\partial}{\partial p'}\langle p'|\alpha\rangle$.

(ii) $\langle\beta|x|\alpha\rangle = \int \mathrm{d}p'\,\phi_\beta^*(p')\mathrm{i}\hbar\frac{\partial}{\partial p'}\phi_\alpha(p')$,
where $\phi_\alpha(p') = \langle p'|\alpha\rangle$ and $\phi_\beta(p') = \langle p'|\beta\rangle$ are momentum-space wave functions.

(b) What is the physical significance of

$$\exp\left(\frac{\mathrm{i}x\Xi}{\hbar}\right),$$

where x is the position operator and Ξ is some number with the dimension of momentum? Justify your answer.

Notice that the partial derivative and integral are smaller when used in a sentence compared with when you're working in a math environment like `\begin{equation} \end{equation}`. If you want to display the full size of such commands in a sentence, you must use the command `\displaystyle{}`, like it's shown here:

Task 4: Your title

This is an example problem taken from **Sakurai2020**:

(a) Prove the following

(i) $\displaystyle\langle p'|x|\alpha\rangle = \mathrm{i}\hbar\frac{\partial}{\partial p'}\langle p'|\alpha\rangle$.

(ii) $\displaystyle\langle\beta|x|\alpha\rangle = \int \mathrm{d}p'\,\phi_\beta^*(p')\mathrm{i}\hbar\frac{\partial}{\partial p'}\phi_\alpha(p')$,

where $\phi_\alpha(p') = \langle p'|\alpha\rangle$ and $\phi_\beta(p') = \langle p'|\beta\rangle$ are momentum-space wave functions.

(b) $\cdots$

I use the package `physics` which provides a great variety of commands for common operations and symbols. For instance, instead of typing `\dfrac{\partial x}{\partial t}`, the `physics` package provides the command `\pdv{x}{t}` which gives the same result. I also defined my own commands, so you can take a look in the `commands.tex` file if you like. I'd also suggest to create a folder and work each problem in a separate `.tex` file. I already included such folder in the `Overleaf` template, but you won't see it if you download the

`Github` template.