HAW Hamburg
Prof. Dr. Franz Korf/

Topic: Graph structures in Rust

Deadline: At the beginning of the third exercise session

This exercise contains two subtasks.

Subtask 1: During exercise 3 to 6, teams of four person will work on a small self-defined project in Rust.

- Put together your team of four persons until your second exercise.
- Create a git-repo with a descriptive name on our git server. Please grant Silke Behn and me read-only access to this repo.
- Think about what you want to work on in the small project. You are welcome to access other crates or services, as long as the project still contains a relevant amount of Rust code written by the team.

This task is not labor-intensive, but it will take some time until you have put together your team and found an interesting project of the right size. **Hence, start this subtask as soon as possible.**

Subtask 2: Implementation of a sorted doubly linked list

You are certainly familiar with this data structure. You can find a description of an unsorted doubly linked list at https://en.wikipedia.org/wiki/Doubly_linked_list. Of course, other data structures implement the task that a list should fulfill very efficiently. But in the current exercise you implement a sorted doubly linked list to exercise the handling of data structures that have cyclic references in Rust.

To fulfill this exercise, you should first work on the following Rust topics:

- The use of **`RefCell<T>`**
- The use of **`struct`**
- The Rust **`impl`** statement, which binds functions / methods to a **`struct`**.
- The use of **`Drop`** trait. Please note that you only need to know the basics of the Rust trait concept for this exercise. (see page 304 of the book Blandy, J., & Orendorff, J. (2021). Programming rust. O'Reilly Media, Incorporated).

The page https://rtoch.com/posts/rust-doubly-linked-list/ (the quality of which is not checked up to now) offers a first introduction to this topic. Please note that intelligent copy-paste will only lead to limited success in this case. Concerning the module exam, the discussion of this exercise and the upcoming project, you must be able to understand and apply these central concepts.

Minimum requirements for the sorted doubly linked list:

- The type of list elements is defined by a type-parameter `<T>`. `T` must implement the `Ord` trait.
- List nodes are stored on the heap. The metadata of a list, i.e. the data that describe the list itself, should be located on the stack.
- Implement typical methods of sorted lists.
- Please document the user defined types. Represent an empty list and a list containing two elements graphically.
- Please implement suitable test functions.