Lab/Discussion 1

EXERCISE 1. Explore the class PYLIST defined in the lectures:

- **(i)** Create lists using the PYLIST constructor with different elements and different sizes.
- **(ii)** Explore the methods defined on PYLIST objects with different arguments.
- **(iii)** Define a sequence of method calls on your example objects such that allocate and deallocate methods will be executed.

EXERCISE 2. Create a class SPYLIST of lists with elements that have a fixed record structure, for which you can explore dictionaries in **Python**.

- **(i)** Modify the PYLIST methods to deal with the structured lists, i.e. for any modification you have to check that the new list element has the correct type.
- **(ii)** Modify the class further using an attribute of the record structure as key, i.e. a list cannot contain two records with the same value of the key attribute.
- **(iii)** Explore how the complexity of *set_item*, *insert* and *append* could be optimised. Discuss the differences between explorative search in the list and binary search after sorting the list elements.

EXERCISE 3. Continue the previous exercise with a new *project* method on SPYLIST.

- **(i)** A projection operation on a record removes some attributes. Implement a projection operation on structured lists that applies the same projection to all list elements.
- **(ii)** Implement a modified projection method, where it is required that all elements of the resulting list are distinct.
- **(iii)** Analyse the complexity in both cases and discuss how sorting can be used to improve complexity.

EXERCISE 4. Continue the previous exercises with a new *equi-join* method on SPYLIST.

- **(i)** For two records $r_1 = (a_1 : v_1, \dots, a_n : v_n)$ and $r_2 = (b_1 : u_1, \dots, b_m : u_m)$ provide pairwise different indices $\{i_1, \dots, i_k\}$ and $\{j_1, \dots, j_k\}$ for some $k \leq \min(n, m)$. The records r_1 and r_2 can be *joined* iff $a_{i_x} = b_{j_x}$ holds for all $1 \leq x \leq k$. In this case the *equi-join* $r_1 \bowtie r_2$ is the combined record of r_1 and r_2 with $n + m$ attributes.

  Extend the equi-join to two structured lists resulting in the list of all joins of records r_1, r_2 from the two lists.
- **(ii)** Analyse the complexity of the *equi-join* method and discuss how sorting can be used to improve complexity.