Lab Discussion 2

Exercise 1. Explore the class definitions AList and DList using unbounded arrays and doubly linked listsa, respectively:

(i) Create lists using the constructors with different elements and different sizes.

(ii) Explore the functions defined on AList and DList objects with different arguments.

(iii) Define a sequence of fiunction calls on your example objects for AList such that allocate and deallocate methods will be executed.

(iv) Create really huge lists so that it becomes difficult to execute the list functions.

(v) Define a modified class SList, which represents singly linked lists with an additional pointer from the dummy node to the last node of the list. Repeat (i) – (iv) for objects of this class.

Exercise 2. Create a derived class RList of lists with elements that have a fixed record structure, e.g. recording students with first name, last name and student_id.

(i) Modify the AList functions 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 (e.g. student_id), 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.

Exercise 3. Continue the previous exercise with a new *project* method on RList.

(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. Explore the idea of the *splice* method either on AList, DList or SList.

Show how to express other operations using the *splice* method. If necessary, modify the implementation of the *splice* method.