F28PL1 Programming Languages
Laboratory 8

A)

i) write a function to drop the first n elements of a list l:

```
drop n l : int -> a′ list -> a′ list
e.g. drop 3 [1,2,3,4,5] ==> [4,5]
```

- to drop 0 elements from a list return the list
- to drop n elements from the empty list, return the empty list
- to drop n elements from a list, drop n-1 elements from the tail

ii) write a function to take the first n elements of a list l:

```
take n l : int -> ‘a list -> ‘a
e.g. take 3 [“a”,”b”,”c”,”d”,”e”] ==> [“a”,”b”,”c”]
```

- to take 0 elements from a list return the empty list
- to take n elements from the empty list, return the empty list
- to take elements from a list, put the head of the list onto the result of taking n-1 elements from the tail

iii) write a function to check if list l1 starts list l2:

```
starts l1 l2 : ‘’a list -> ‘’a list -> bool
e.g. starts [1,2,3] [1,2,3,4] ==> true
```

- an empty list starts a list
- a list does not start an empty list
- a list starts a second list if they have same head and the tail of the first starts the tail of the second

B)

iv) write a function to check if list l1 is contained in list l2:

```
contains l1 l2 : ‘a list -> ‘a list -> bool
e.g. contains [“d”,”e”,”f”] [“a”,”b”,”c”,”d”,”e”,”f”,”g”,”h”]
==> true
```

- a list is not contained in an empty list
- a list is contained in a second list if it starts the second list
- otherwise, a list is contained in a second list if it is contained in the tail of the second list

v) write a function to delete a list from another list:

```
delete l1 l2 : ‘’a list -> ‘’a list -> ‘’a list
e.g. delete [3,4,5] [1,2,3,4,5,6] ==> [1,2,6]
```

- to delete a list from the empty list, return the empty list
- to delete a list from a second list, if the first list starts the second list then drop the length of the first list from the second list
- otherwise, put the head of the second list onto the result of deleting the first list from the tail of the second

vi) write a function to delete every occurrence of a list from another list:

```
deleteAll l1 l2 : ‘’a list ->’’a list -> ‘’a list
e.g. deleteAll [1,2,3] [3,2,1,2,3,2,1,2,3] ==> [3,2,2]
```

- all occurrences have been deleted from an empty list
- if the first list starts the second list, delete it and then delete all occurrences in the remaining list
- otherwise, put the head of the second list on the front of deleting all occurrences of the first list in the tail