Εργασία 1 (Question 2)

Ονοματεπώνυμο: Βλάσης Μαξούτης
A.M: 11152400118

Below are listed the time complexities of the functions related to doubly-linked lists in the corresponding, that were implemented in the corresponding C module in the same directory as this PDF.

Create()

Time complexity: O(1). It is constant, since it simply calls malloc to allocate memory for a struct of type list and initialize its members.

Size()

Time complexity: O(1). Returns the "count" member of a struct list, which is done in constant time.

IsEmpty()

Time complexity: O(1). Returns whether or not the "count" member of a list is 0 or not, which is done in constant time.

GetFirst()

Time complexity: O(1). Returns a pointer to the head of a list, which is done in constant time since it is stored as a member of the struct list.

GetLast()

Time complexity: O(1). Returns a pointer to the tail of a list, which is done in constant time since it is stored as a member of the struct list.

GetNode()

Time complexity: O(n). Worst case we have to loop over the entire list to find the node, so it is done in linear time.

GetPrev()

Time complexity: O(1). It is done in constant time since the previous and next node of a given node are stored in the corresponding struct of a list node.

GetNext()

Time complexity: O(1). It is done in constant time since the previous and next node of a given node are stored in the corresponding struct of a list node.

AddBefore()

Time complexity: O(1). Constant time because similiarly to Create(), it creates a node, and then it adds it right before the given node, which can be done in constant time since we store the pointer to that node.

AddAfter()

Time complexity: O(1). Similiar to AddBefore(). Constant time because similiarly to Create(), it creates a node, and then it adds it right after the given node, which can be done in constant time since we store the pointer to that node.

AddFirst()

Time complexity: O(1). Constant time because it creates a node and adds it as the head of the list, for which we have stored a pointer

AddLast()

Time complexity: O(1). Constant time because it creates a node and adds it as the tail of the list, for which we have stored a pointer

Remove()

Time complexity: O(n). Linear time since we have to loop through the entire list in order to find all nodes that have the given value "i".

Print()

Time complexity: O(n). Linear time since we have to loop through the entire list in order to print each node.