Data Structure and Algorithms **Computer Engineering Degree** **University Carlos III of Madrid** **COURSE 2020-2021**	uc3m | Universidad Carlos III de Madrid

Name:

Instructions:

- Exam duration: 60'.
- Download the partial1.zip and unzip it .
- In the file midtermexam.py you must write your solution. Also, add a python comment at the beginning of the file with your first and last name.
- DO NOT MODIFY the test.py file (it helps you test your solution).
- spyder or pycharm) will be open on your computer . You cannot consult any material (code, notes, books, etc).
- When there are 5 minutes left, the teacher will notify you to upload your solution. You will be able to open the browser and enter your small global classroom group. Upload only the midtermexam.py file to the “MidTermMockExam” activity in Aula Global.
- It is your responsibility to check that you have uploaded the correct version of the file.
- Cell phones must be turned off and can never be on the table.
- You cannot leave the classroom before 30 minutes have passed.
- It is not allowed to go to the bathroom.

Problem: In the MyDList class , you complete the *removeSmaller (e) function* , which receives an integer e. The function must remove all nodes whose elements are less than e . The function returns nothing. The function modifies the list.
Here are some examples:

list before operation	operation	list after operation
9<->4<->4<->6<->3<->12<->2<->1	l. *removeSmaller* (1)	9<->4<->4<->6<->3<->12<->2<->1
9<->4<->4<->6<->3<->12<->2<->1	l. *removeSmaller* (2)	9<->4<->4<->6<->3<->12<->2
9<->4<->4<->6<->3<->12<->2<->1	l. *removeSmaller* (6)	9<->6<->12
9<->4<->4<->6<->3<->12<->2<->1	l. *removeSmaller* (12)	12
9<->4<->4<->6<->3<->12<->2<->1	l. *removeSmaller* (20)	empty list
(empty list)	l. *removeSmaller* (1)	(empty list)

You cannot add new attributes or functions to classes DNode and MyDList .
For the solution to be considered correct, the proposed function must be **correct** (it solves the problem), **robust** (has no errors and works for any input) and **efficient** in terms of temporal and spatial complexity (avoid the use of auxiliary structures). Also, the code should be easy to understand and maintain.
The use of Python structures such as dictionaries or lists is not allowed.