REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université de M'sila
Faculté des Mathématiques et de l'Informatique
Département d'informatique

جامعة المسيلة
كلية الرياضيات والإعلام الآلي
قسم الإعلام الآلي

Level: 1st year computer science
Material: ADS2

TD/TP Series No.: 04

Academic year: 2023/2024
Chapter 3 : linked Lists

Note Use the **new** and **delete operator** and the & reference pass (C++)

Exercise 1: (TD)

Write the "**get**" function that returns the value of the item at location 'i'. If the location doesn't exist, it exits the program using "**exit(-1)**".

Exercise 2: (TP)

Write the "**insert**" function that adds an item to the list at the specified location. If the location is less than 1, it adds it to the beginning of the list; if it is larger than the size of the list, it adds it to the end of the list.

Exercise 3: (TD)

Write the **array2list** function that populates a list from an array.

Exercise 4: (TP)

Write the **contains** function that tells us whether a number is in the list or not.

Exercise 6 : (TD)

Write the **reverseSorted** function that adds an item to a list **sorted** in **ascending** order while maintaining the sort order.

Exercise 7: (TP)

Write the **reverseList** function that inverts a list.

Exercise 8: (TD/TP)

Write the **rmvDuplicate** function that removes all duplicates from a **sorted queue** and leaves only one occurrence of each item.

Exercise 9: (TD/TP)

Write the **reverseArray** function, which inverts an array using a **stack**.

Exercise 10: (TD/TP)

Write the duplicateList function, which duplicates items in a list using a **queue**.

Exercise 10: (at home)

Consider the following structures:

```
typedef struct Cell {
  int data;
  struct Cell* next;
} Cell;
typedef struct List {
  int size;
  Cell* head;
} List;
```

Rewrite the following functions:

- `List *initList()`
- `Bool add_head(List &lst, int d)`
- `bool append_end(List &lst, int d)`
- `bool delete_head(List &lst)`
- `bool delete_end (List &lst)`