```
class Node
{
public:
	int key;
	Node* next;
};
```

```
Node* newNode(int key)
{
	Node* node = new Node;
	node->key = key;
	node->next = nullptr;
	return node;
}
```

Use the previous codes in the following tasks:

1. Create a function (`fillVect`) that uses the method (`push_back`) to fill n elements in a vector (`vector <int> v`).
2. Create a recursion function (`CreateListFromVector`) to generate a list (`Node* L`) of "n" entries from the vector(created in task 1).
3. Create a recursion function (`printList`)to print list elements (print the values of the created list).
4. Create a recursion function (`max`) to get the list's maximum value.
5. Create a recursion function (`elementPosition`) that returns the position of an element (`int e`) in a List (`Node* L`) if it exists and -1 otherwise.
6. Create a recursion function(`addEnd`) to append(add at the end) an element to an existing list.
7. Create a recursion function(`removePelemtn`) to remove an element at position p.
8. Create a recursion function(`reverseList`) that returns the list's reverse.
9. Create a recursion function(average) that computes the list elements' average.
10. Create a recursion function(sum) that computes the list elements' sum.

Utilisez les codes précédents dans les tâches suivantes :

1. Créez une fonction (fillVect) qui utilise la méthode (push_back) pour remplir n éléments dans un vecteur (vecteur <int> v).
2. Créer une fonction récursive (CreateListFromVector) pour générer une liste (Node* L) de "n" entrées à partir du vecteur (créé dans la tâche 1).
3. Créer une fonction récursive (printList) pour imprimer les éléments de la liste (imprimer les valeurs de la liste créée). 4.
4. Créez une fonction récursive (max) pour obtenir la valeur maximale de la liste.
5. Créer une fonction récursive (elementPosition) qui retourne la position d'un élément (int e) dans une liste (Node* L) s'il existe et -1 sinon.
6. Créer une fonction récursive (addEnd) pour ajouter (ajouter à la fin) un élément à une liste existante.
7. Créer une fonction récursive (removePelemtn) pour supprimer un élément à la position p. 8.
8. Créer une fonction récursive (reverseList) qui retourne l'inverse de la liste. 9.
9. Créer une fonction récursive (average) qui calcule la moyenne des éléments de la liste. 10.
10. Créez une fonction récursive (sum) qui calcule la somme des éléments de la liste.