class Node: def **init**(self, valeur, suivant: 'Node' =None): self.valeur = valeur self.suivant = suivant

class Liste: def **init**(self, head: Node): self.head = head

```
def __str__(self) -> str:
    return "<" + ",".join(map(str, self.to_list())) + ">"

def to_list(self) -> list:
    if self.head is None:
        return []
    return [self.head.valeur] + self.cdr().to_list()

def __len__(self) -> int:
    """Longueur de la liste.
    """
    if self.head is None:
        return 0
    return 1 + len(self.head.suivant)

def append(self, valeur) -> None:
    """Ajouter une valeur à la fin de la liste.
    """
    # cas de base :
    # on doit s'arrêter dès que la liste est terminer
    if self.head.suivant is None:
        # on créé un nouveau Node
        self.head.suivant = Node(valeur)
    # récursion pour atteindre la fin de la liste
    self.cdr().append(valeur)

def car(self):
    """Récupérer la valeur en tête de liste
    """
    return self.head.value

def cdr(self) -> 'Liste':
    """Récupérer la liste sans son premier élément.
    """
    if self.head is None:
        return None
    return Liste(self.head.suivant)

def last(self):
    """Récupérer le dernier élément de la liste.
    """
    if self.head.suivant is None:
```

```
        return self.last(Liste(self.head.suivant))
    return self.head.valeur
```

1. Quelle est l'instruction pour créer une liste vide nommée `p` ?

p = None

2. Quelle est l'instruction pour créer une liste d'entiers p ne contenant

qu'un seul élément (par exemple la valeur 1) ?

p = Node(1)

3. Quelles sont les instructions pour créer une liste d'entiers contenant

exactement les valeurs 1, 2 et 3 dans cet ordre ?

p = Node(1, Node(2, Node(3)))

4. On considère désormais la liste `L = (1(23)(45(678)))`. Comment récupérer

les valeurs ou listes suivantes : 1, (3), 3, (5(678)) et enfin 6 ?

L = Node(1, Node(Node(2, Node(3)), Node(4, Node(5, Node(Node(6, Node(7, Node(8)))))))

print(L.valeur) print(Liste(L.suivant.valeur.suivant)) print(Liste(L.suivant.valeur.suivant.valeur))