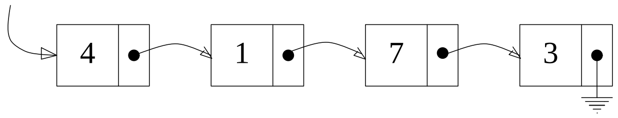
Listes chaînées par pointeurs...

On considère les déclarations suivantes :

```
typedef struct cellule
{ int Val;
  struct cellule * PtrSuiv;
} Cellule;

typedef Cellule * PtrCellule;
```



Exemple d'une liste chaînée.

▸ **Exercice 1**. Écrire une fonction de nom `CreerCellule()` créant une nouvelle cellule et retournant son adresse. La fonction prendra en paramètre la valeur `val` qui initialisera le champ `Val`.

▸ **Exercice 2**. Écrire une fonction de nom `CreerListe()` de création d'une liste chaînée d'entiers non nuls saisis au clavier. On arrêtera la saisie lorsque l'utilisateur entrera l'entier 0.

▸ **Exercice 3**. Donner une fonction de nom `AffListe()` permettant d'afficher une liste chaînée d'entiers dans l'ordre du chaînage.

▸ **Exercice 4**. Donner une fonction de nom `AffRevListe()` d'affichage d'une liste chaînée d'entiers dans l'ordre inverse du chaînage.

▸ **Exercice 5**. Écrire une fonction de nom `RevListe()` permettant d'inverser l'ordre des éléments d'une liste chaînée d'entiers. La fonction créera une nouvelle liste totalement indépendante de la liste chaînée d'origine, et ne devra surtout pas détruire cette liste chaînée d'origine.

▸ **Exercice 6**. Écrire une fonction de duplication d'une liste chaînée. La fonction créera une liste indépendante de la liste chaînée initiale.

Exercices supplémentaires

▸ **Exercice 7**. Refaire l'exercice précédent en inversant la liste sur place. Le chaînage initial sera donc détruit.