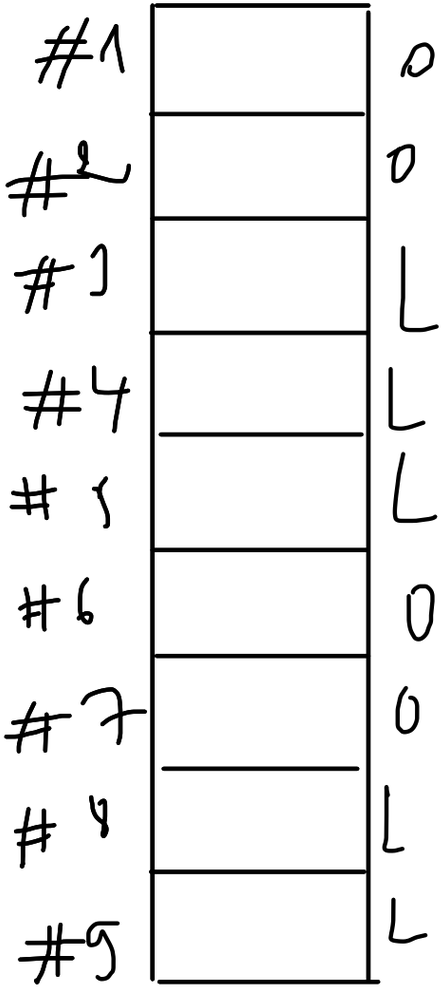
chapitre 4. Liste simplement chaînée

On veut stocker les éléments suivants dans la mémoire T[]={2,6,7,9},
proposer la structure adéquate dans les cas suivants

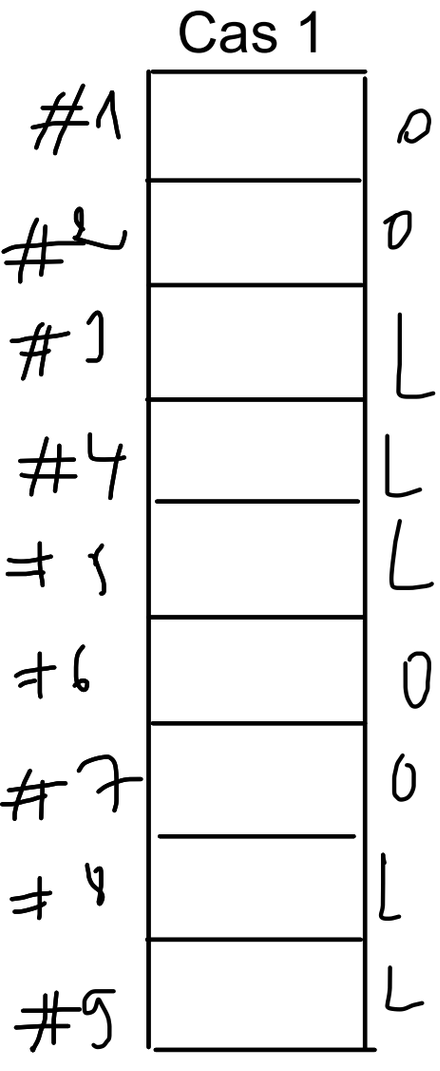
Cas 1



Cas 2

Cas 3

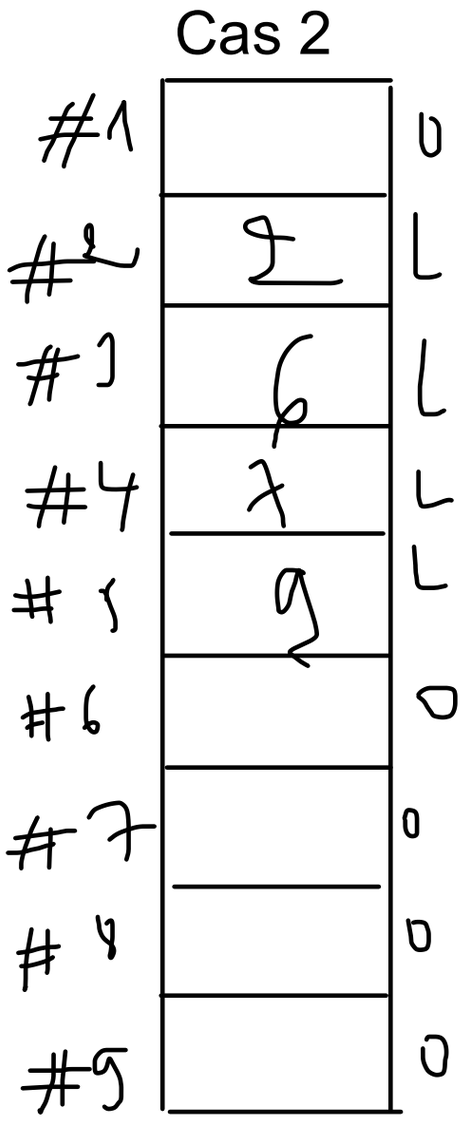
Cas 4



Non

il n'y pas 4 cas

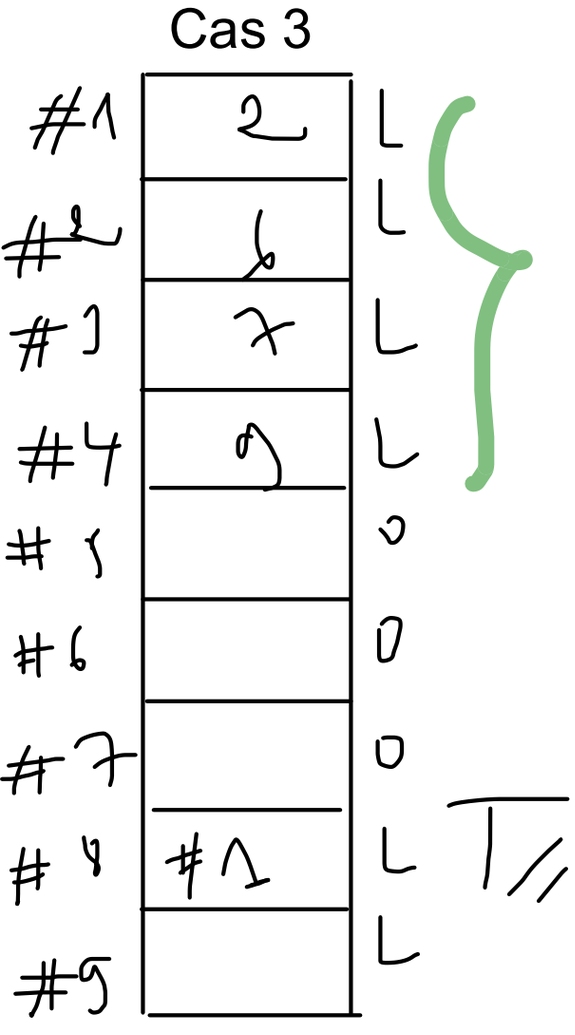
Libre et suivi



Ici, on peut utiliser

un tableau

int T[4]



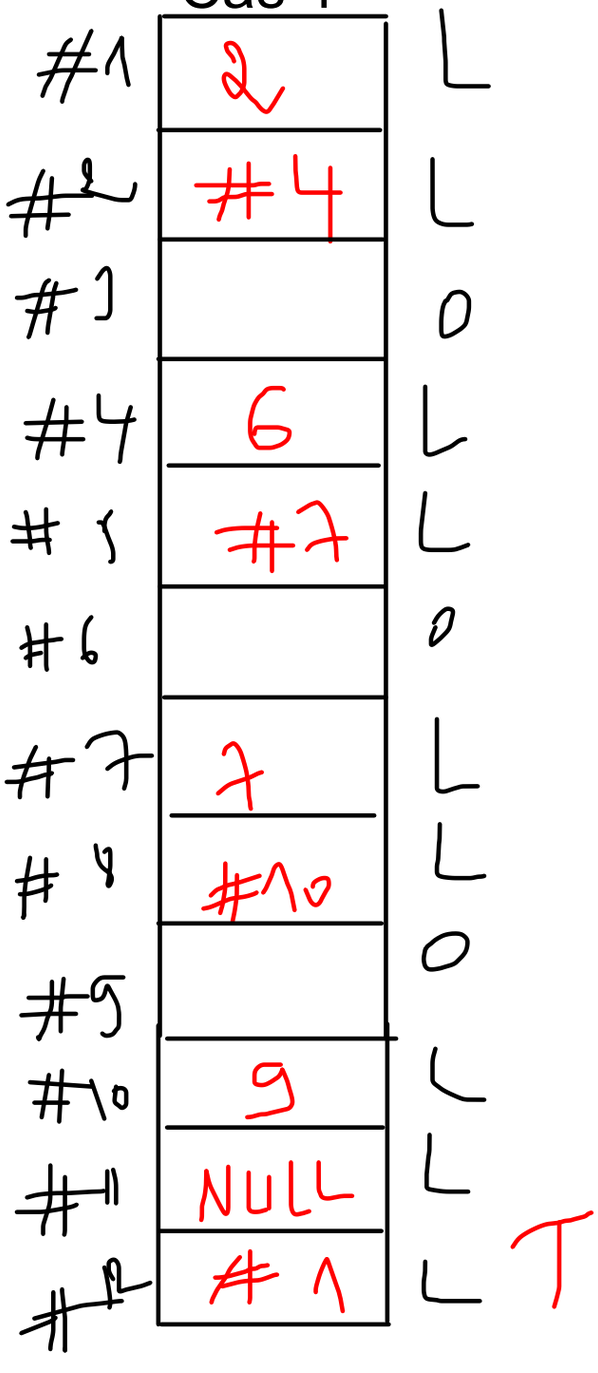
Solution1. on utilise un tableau statique

```
int T[4];
```

solution 2. on uilise un tableau dynamique

```
int *T;
T=(int *) malloc ( 4 * sizeof(int));
```

Cas 4



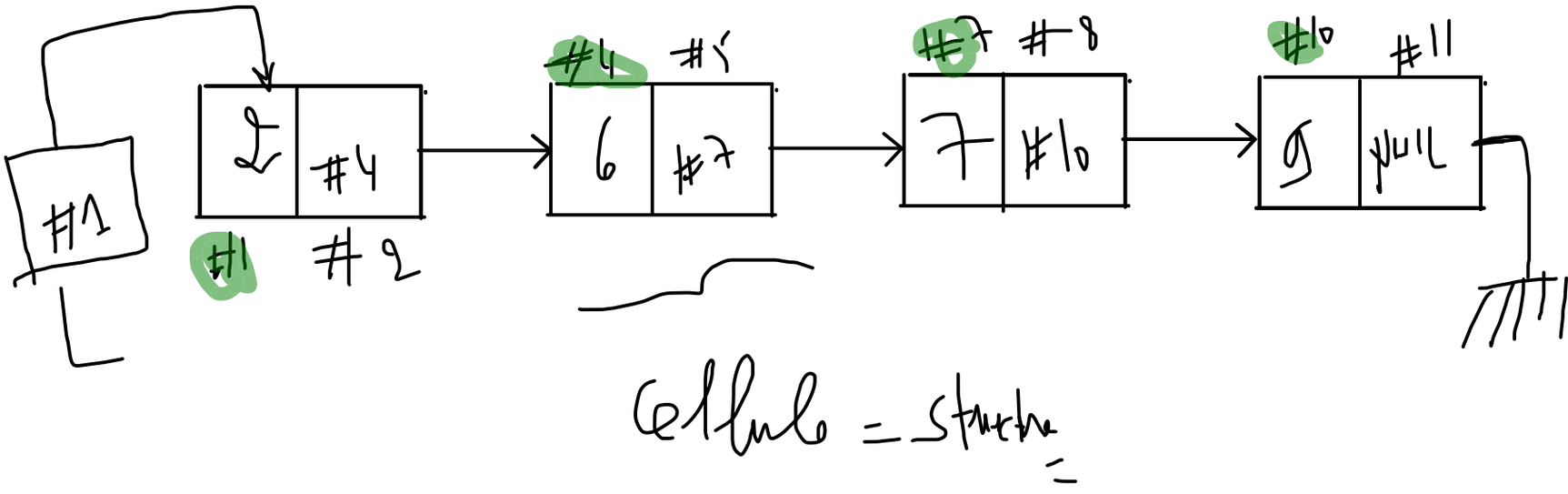
T = ⟨2, 6, 7, 9⟩

Cas 4

#1	2	L
#2	#4	L
#3		0
#4	6	L
#5	#7	L
#6		0
#7	7	L
#8	#10	L
#9		0
#10	9	L
#11	NULL	L
#12	#1	L

T

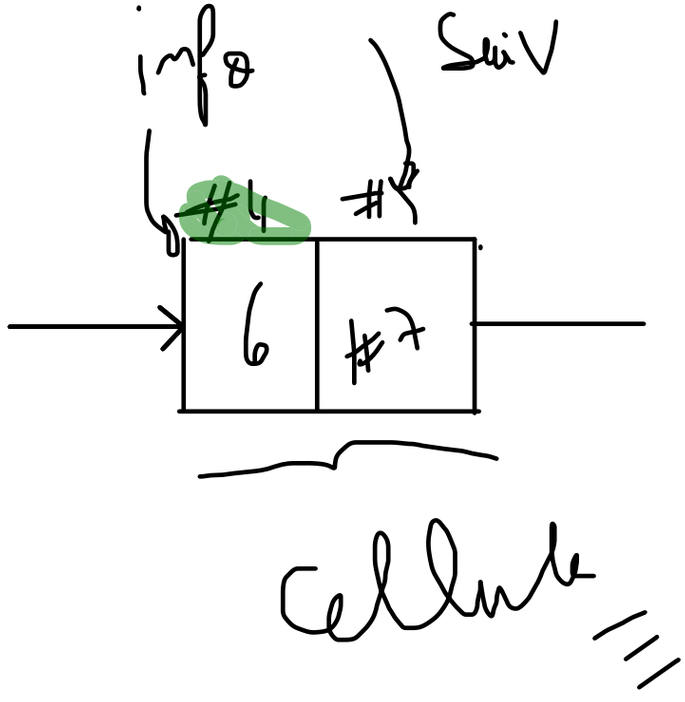
L on l'appelle liste simplement chaînée



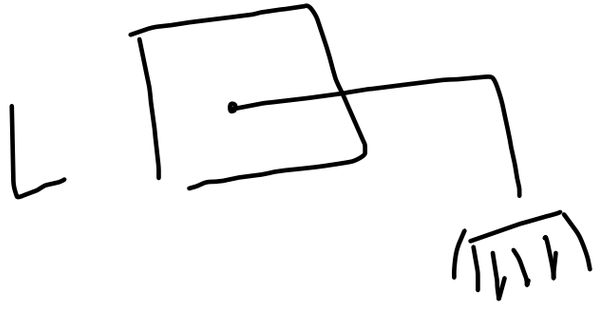
Cellule = Structure

Déclaration d'une liste chaînée

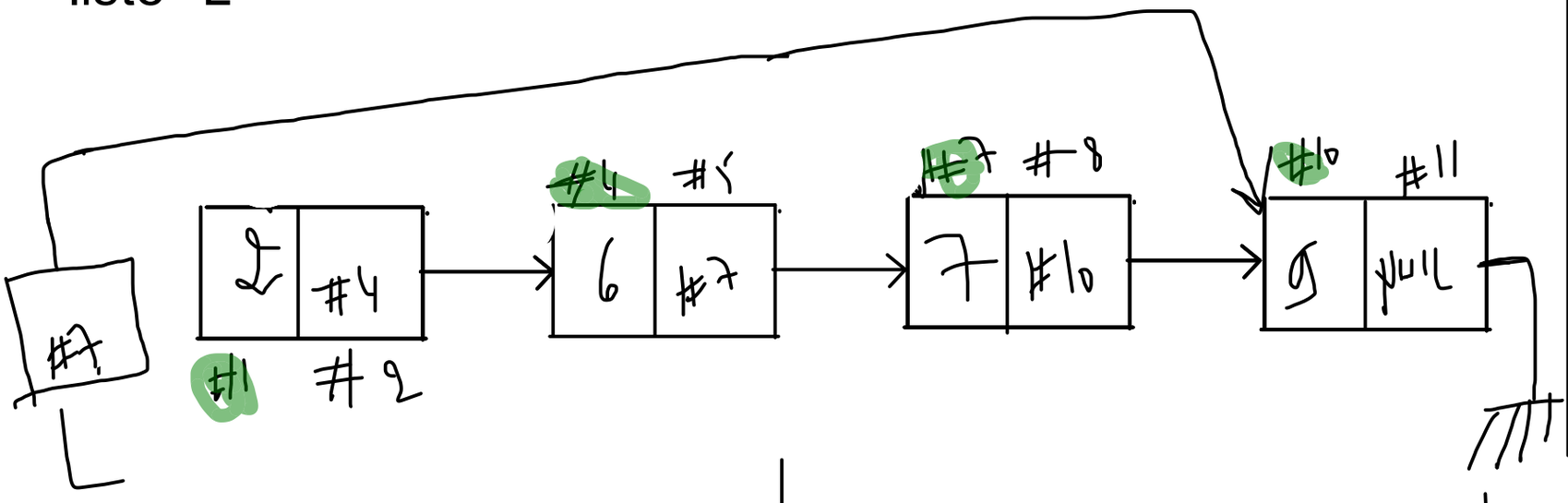
```
typedef struct cellule{
int info;
struct cellule *suiv;
}liste
```



liste *L;



liste *L



L = L → suiv

L → info = 9

L → suiv = NULL

L → info = 2

L → suiv = #4

L = L → suiv

L → info = 6

L → suiv = #7

L = L → suiv

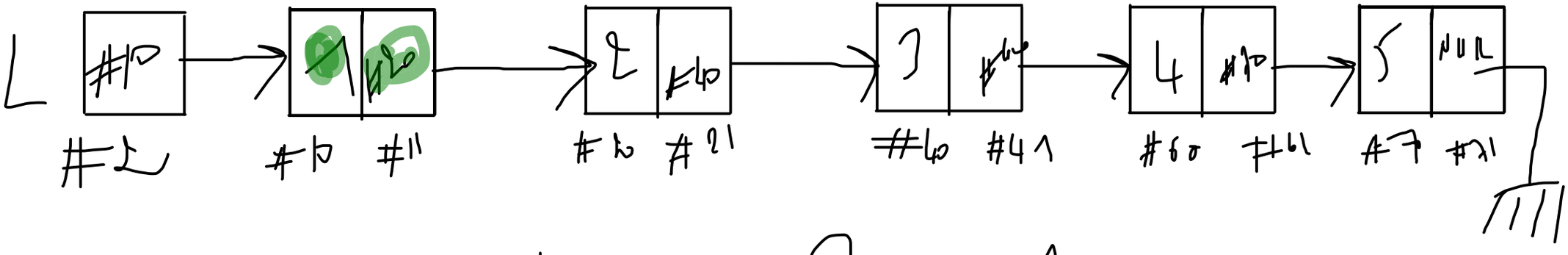
L → info = 7

L → suiv = #10

```
typedef struct cellule{
int info;
struct cellule *suiv;
}liste
```

Défi=Une liste chaînée est un pointeur sur la première cellule

liste *L



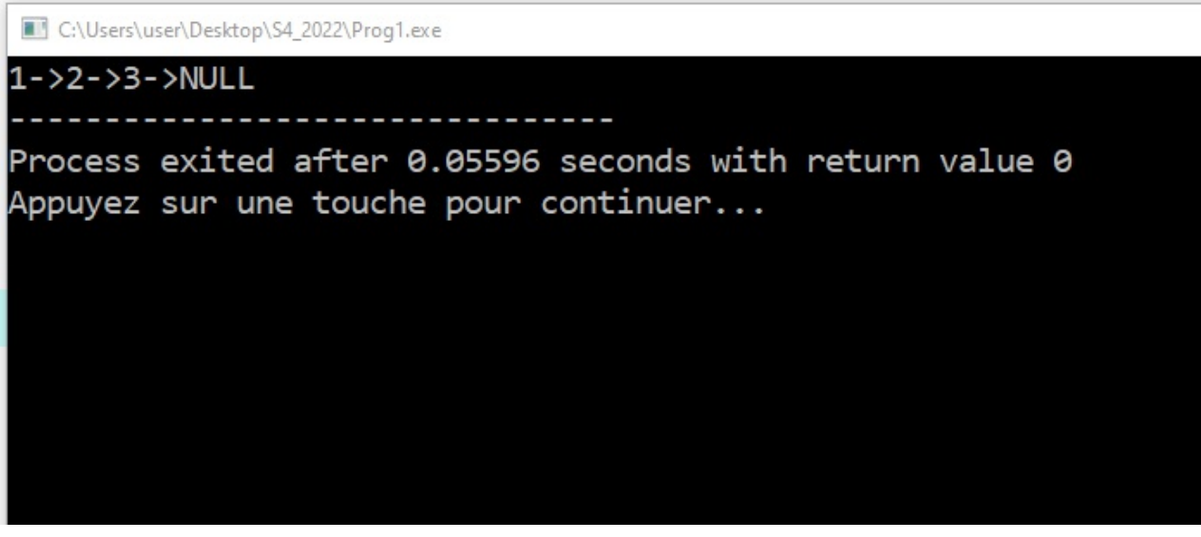
L → info = 1

L → suiv = #20

```
    //Branchement
    C->suiv=L;
    L=C;
    return L;
}
///Afficher liste
void AfficherListe(liste *L)
{
    while(L!=NULL)
    {
      printf("%d->",L->info);
      L=L->suiv;
    }
    printf("NULL");
}
//PP
main()
{
    liste *L=NULL;
    L=AjoutDebut(L,3);
    AfficherListe(L);
}
```

```
Sélection C:\Users\user\Desktop\S4_2022\Prog1.exe
3->NULL
--------------------------------
Process exited after 0.04711 seconds with return value 0
Appuyez sur une touche pour continuer...
```

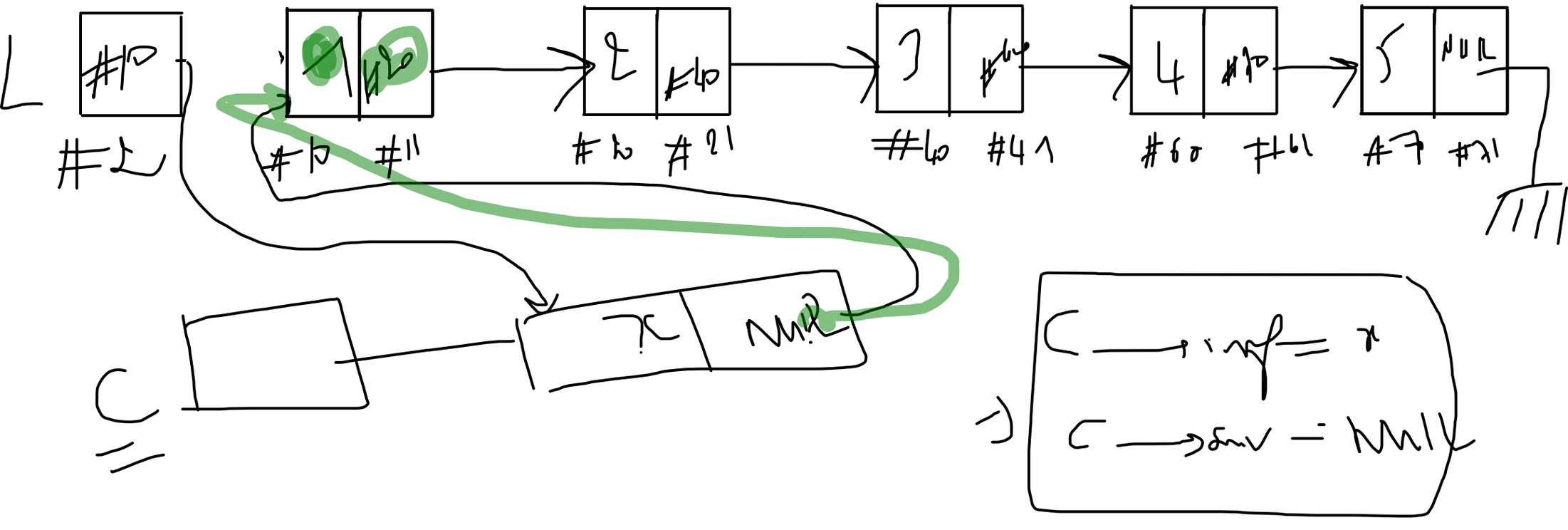
```
}
//PP
main()
{
    liste *L=NULL;
    L=AjoutDebut(L,3);
    L=AjoutDebut(L,2);
    L=AjoutDebut(L,1);
    AfficherListe(L);
}
```



Ajout début

```
liste *c;
c=(liste*)malloc(sizeof(liste))
```

liste *L



```
liste *C;
    //Réservation de l'espace
    C=(liste*)malloc(sizeof(liste));
    if(C!=NULL)
    {
        C->info=x;
        C->suiv=NULL
    }
```