Création d'une liste chaînée :

En général, pour créer une liste, on crée un maillon et ensuite un « outil » pour le manipuler, car, rappelez-vous, lorsqu'on a un maillon (plus précisément le maillon de tête), on peut reconstruire toute la liste.

1. Méthode 1 :

Cette méthode consiste à créer le maillon d'abord, puis à créer un pointeur sur le maillon. Ce dernier représente toute la liste.

Un exemple simple.

```
struct ListeNoeuds {
        int data ;
        struct ListeNoeuds *suivant;
};

struct ListeNoeuds *tete ;
```

Un autre exemple.

```
typedef struct Fiche {
        int identifiant;
        char nom[20];
        char prenom[20];
        int tel[10];
        struct Fiche *suivant;
}Contact;
Contact* listeContacts = NULL;
```

Un exemple particulier.

```
typedef struct maillon {
        int info;
        struct maillon *succ;
        struct maillon *pred;
} maillon;
typedef maillon *liste;
```

2. Méthode 2 :

Une autre méthode consiste à créer le maillon d'abord, puis à créer une structure contenant divers éléments de manipulation qui représente la liste.

```
typedef struct element *Pelement;

typedef struct element{
        int x;
        Pelement suivant;
}Element;

typedef struct liste{
```

Pelement premier;
Pelement courant;
Pelement dernier;
}Liste;

N.B. : Il est extrêmement important de bien choisir la définition de la liste chaînée car d'elle dépendent les déclarations, définitions et manipulations suivantes.

Un exemple de bout en bout :

```c
#include <stdio.h>
#include <stdlib.h>

// Définition de la structure ListeNoeuds
struct ListeNoeuds {
    int data;
    struct ListeNoeuds *suivant;
};

// Fonction pour créer un nouveau nœud
struct ListeNoeuds* creerNoeud(int data) {
    struct ListeNoeuds* nouveauNoeud = (struct ListeNoeuds*)malloc(sizeof(struct Li
    nouveauNoeud->data = data;
    nouveauNoeud->suivant = NULL;
    return nouveauNoeud;
}

int main() {
    // Initialisation des nœuds
    struct ListeNoeuds *tete = creerNoeud(10);
    tete->suivant = creerNoeud(20);
    tete->suivant->suivant = creerNoeud(30);

    // Affichage des données de la liste chaînée
    struct ListeNoeuds *temp = tete;
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->suivant;
    }
    printf("NULL\n");

    return 0;
}
```

Définition de la structure ListeNoeuds

```
struct ListeNoeuds {
    int data;
    struct ListeNoeuds *suivant;
};
```

Fonction pour créer un nouveau nœud

```
struct ListeNoeuds* creerNoeud(int data) {
    struct ListeNoeuds* nouveauNoeud = (struct ListeNoeuds*)malloc(sizeof(struct
ListeNoeuds));
    nouveauNoeud->data = data;
    nouveauNoeud->suivant = NULL;
    return nouveauNoeud;
}
```

Initialisation des nœuds

```
    struct ListeNoeuds *tete = creerNoeud(10);
    tete->suivant = creerNoeud(20);
    tete->suivant->suivant = creerNoeud(30);
```

Affichage des données de la liste chaînée

```
    struct ListeNoeuds *temp = tete;
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->suivant;
    }
```