Le C —
Types complexes et listes

https://tinyurl.com/syjh3j8z

Nathanael Demacon

Listes chaînées

Une liste chaînée est une structure de données utilisée pour agrandir dynamiquement la mémoire au fur et à mesure d'ajouter de nouveaux éléments.


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

typedef struct Node Node;
struct Node
{
    int data;
    Node *next;
};

Node *ll_push_front(Node *first, int data)
{
    Node *new = malloc(sizeof(Node));
    new->data = data;
    new->next = first;

    return new;
}
```

```c
void ll_free(Node *first)
{
    if (first != NULL)
    {
        ll_free(first->next);
        free(first);
    }
}

void ll_print(Node *first)
{
    while (first != NULL)
    {
        printf("%d", first->data);
        first = first->next;

        if (first != NULL)
        {
            printf(", ");
        }
    }
}

int main()
{
    Node *students = NULL;
    students = ll_push_front(students, 1);
    students = ll_push_front(students, 2);

    ll_print(students); // 2, 1
    ll_free(students);

    return 0;
}
```

Les énumerations

Une énumération permet de donner un nom à des entiers pour rendre le code plus lisible et maintenable.

Énumérations
Déclaration

Pour déclarer une énumération, nous utilisons le mot clé “enum” suivi de son nom puis de ses valeurs. Par exemple pour les jours de la semaine :

```
enum WeekDay
{
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday,
    Sunday
};
```

Énumérations
Déclaration

Pour déclarer une variable de type enum, par défaut il faut rajouter “enum” devant le nom du type.

```c
#include <stdio.h>

enum WeekDay
{
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday,
    Sunday,
};

int main()
{
    enum WeekDay day = Friday;
    printf("%d\n", day); // 4
    return 0;
}
```

On remarque que “Friday” est directement accessible sans avoir à préciser qu’il vient de l’énumération “WeekDay”.

Énumérations
Déclaration

Les valeurs entières d'une énumération commencent par "0" mais il est possible de les changer manuellement. Les valeurs suivantes non affectées s'incrémentent automatiquement.

```
enum WeekDay
{
   Monday,         // 0
   Tuesday,        // 1
   Wednesday = 5, // 5
   Thursday,       // 6
   Friday,         // 7
   Saturday,       // 8
   Sunday,         // 9
};
```

Énumérations

Typedef

Tout comme pour les structures, c'est un peu pénible d'avoir à préciser "enum" à chaque fois. On peut utiliser "typedef" pour créer un alias.

```c
#include <stdio.h>

typedef enum
{
   Monday,
   Tuesday,
   Wednesday,
   Thursday,
   Friday,
   Saturday,
   Sunday,
} WeekDay;

int main()
{
   WeekDay day = Friday;
   printf("%d\n", day); // 4
   return 0;
}
```