INFO525: Examen

January 9, 2026

1 Trace d'exécution/Récursivité (5pts)

Écrivez la trace d'exécution du programme suivant :

```
void syracuse_it(int n){

        while(n > 1){
                printf("%d|", n) ;
                if( (n % 2) == 0)
                        n = n / 2 ;
                else
                        n = (3 * n) + 1 ;
        }
        printf("1\n" ) ;
}
```

Question 1 Écrivez la trace d'exécution de ce programme pour n=64, n=3, n=8 ;

Question 2 Dans quels cas cette fonction se termine-t-elle ?

Question 3 Peut-on savoir si cette fonction se termine quelque soit n ?

Question 4 Écrivez une version récursive de cette fonction.

2 Structure de données (15pts)

2.1 Dictionnaire

Un dictionnaire est une liste dans laquelle chaque élément **est un couple Clé / Valeur**. Par exemple :

```
{ ProductID: 39, UnitPrice: 18, Quantity: 130, Discount: 1 }
```

Dans l'exemple ci-dessus, la clé est une **chaîne de caractère** et la valeur un **entier**. Ce qui donnerai langage C.

```
typedef struct {
        String cle ;
        int val;
} CleVal ;
```

Un dictionnaire doit pouvoir contenir un nombre quelconque d'éléments, vous allez utiliser une liste chaînée pour construire votre type dictionnaire. Le type liste chaînée est défini ci-dessous.

```
typedef int Element ;

typedef struct Cell {
        Element e ;
        struct Cell* next;
} Cell ;

typedef struct LinkedList {
        Cell * tail ;
        Cell * head ;
        int nb ;
} LinkedList ;

/* Initialise le TAD */
void ll_init(LinkedList* ll) ;
/* Ajoute un element en queue ? */
void ll_append(LinkedList* ll , Element e) ;
/* Nombre d e de la ll */
int ll_nbe(LinkedList ll) ;
/* retourne l'element a la position i */
Cell* ll_get(LinkedList* ll , int i) ;
```

Question 1 Écrivez la fonction **ll_get(LinkedList* ll, int i)** quelle est sa complexité ? Est-elle meilleure que d'accéder au i ème élément d'un tableau ?

Question 2 Écrivez la modification à réaliser pour que LinkedList s'applique au type **CleVal**.

Question 3 Écrivez un nouveau type **Dictionnaire** qui contiendra une liste chaînée et le nombre d'élément CléValeur.

Question 4 Écrivez les fonctions de dictionnaire :

```
dico_init(Dictionnaire* d)

dico_aff(Dictionnaire d)

dico_add(Dictionnaire* d, char* cle, int val)
```

Question 5 Modifiez votre fonction **dico_add** pour que l'on ne puisse pas ajouter une clé qui existe déjà. Vous pouvez ajouter des fonctions utiles si nécessaire.

Question 6 Créez une fonction **dico_get(char* cle)** qui retourne la valeur associée à la clé passée en paramètre. Pour cela vous pouvez utiliser la fonction **strcmp** qui retourne 0 si deux chaînes sont identiques.

Question 7 Quelle est la complexité de votre algorithme de la question 5 ?

Question 8 Si les clés étaient triées quel algorithme efficace connaissez vous rechercher une clé ? Donnez sa complexité.