Groupe 01

Exercice 01 Écrire une fonction récursive qui permet de calculer la somme des chiffres d'un entier naturel n.

Solution

```
int sommeChiffre(int n) {
    if(n < 10) return n;
    else return n % 10 + sommeChiffre(n / 10) ;
}
```

Exercice 02 Soit l une liste simplement chainée d'entiers. Écrire une fonction / procédure itérative qui permet d'afficher la liste l par ordre inversé.

Solution

```
void afficheListeInverse(Liste l) {
    Liste p, q;
    p = l;
    q = l;
    while(q->suivant != NULL) q = q->suivant;
    while(p != q) {
        printf("%d ", q->val);
        while(p->suivant != q) p = p->suivant;
        q = p;
        p = l;
    }
    printf("%d ", p->val);
}
```

Groupe 02

Exercice 01 Écrire une fonction récursive qui permet de vérifier si un entier naturel n est premier.

Solution La variable i est initialisée à $n/2$.

```
int premier(int n, int i) {
    if(n == 1) return 0;
    if(i == 1) return 1;
    else
        if(n % i == 0) return 0;
        else return premier(n, i - 1);
}
```

Exercice 02 Soit l une liste simplement chainée. Écrire une fonction qui permet d'inverser les liens de chainages.

Solution

```
Liste inverserListe(Liste l) {
    Liste p, q, r;
    p = NULL;
    r = l;
    while(r != NULL) {
        q = r;
        r = q->suivant;
        q->suivant = p;
        p = q;
    }
    return q;
}
```

Groupe 03

Exercice 01 Écrire une fonction récursive qui permet de calculer la somme

$$s = 1 + 2 + \ldots + n$$

Solution

```
int somme(int n) {
    if(n == 0) return 0;
    else return n + somme(n - 1);
}
```

Exercice 02 Soit l une liste simplement chainée d'entiers. Écrire une fonction qui permet de calculer le nombre d'éléments différents de cette liste.

Solution

```
int nbEltDiff(Liste l) {
    int nb = 0, trouve;
    Liste p;
    p = l;
    while(l != NULL) {
        p = l->suivant;
        trouve = 0;
        while(p != NULL) {
            if(l->val == p->val) trouve = 1;
            p = p->suivant;
        }
        if(trouve == 0) nb++;
        l = l->suivant;
    }
    return nb;
}
```