Université Abou Bekr-Belkaïd – Tlemcen
Faculté des Sciences
Département d'Informatique

Année universitaire 2022-2023
AAC (**A**lgorithmique **A**vancée et **C**omplexité)

TD 2 – *Complexité Temporelle*

Exercice n°01 :

1. Pour chacune des méthodes suivantes, trouver sa ***WCTC*** et sa ***BCTC*** :

```
int nbreChiffres(int N){
        int C = 0;
        if (N < 0)
                return nbreChiffres(N * (-1));
        while(N != 0){
                C++; N = N / 10;
        }
        return C;
}
```

```
void M2(int N, int M){
        int S = 0 ;
        for(int i=1 ; i <= N; i = i * 2){
            for(int j=1 ; j <= M; j++)
                    for(int k=j ; k >= 1; k--){ S++ ; }
        }
}
```

```
void M3(int N, int M){
        int S = 0 ;
        for(int i=1 ; i <= N; i = i + M){
                for(int j=1 ; j <= M; j++){
                        S++ ;
                }
        }
}
```

```
void M5(int N, int M){
        for(int i=1 ; i <= N; i = i + 1){
                for(int j=1 ; j <= M; j= j * 2) S++ ;
        }
        if(N > 0) M5(N - 1, M) ;
}
```

```
void Inverser(int[] T, int D, int F){
        if(D == F)
                System.out.print(T[D] + " ");
        else {
                int M = (D + F)/2;
                Inverser(T, M + 1, F);
                Inverser(T, D, M);
        }
}
```

```
void M1(int N) {
        int i = 1, j = 1 ;
        while (i ≤ N){
                j = 1;
                while(j ≤ i){ j++ ; }
                i++ ;
        }
}
```

```
//T est un tableau trié en ordre croissant
int ElementDominant(int[] T){
        int fmax = 0, e, i=0, f, j ;
        while(i < T.lenght){
                f = 0, j = i ;
                while(T[i] == T[j]){ f++ ; j++;}
                if(f > fmax){fmax = f ; e = T[i];}
                i++ ;
        }
        return e ;
}
```

```
void M4(int N, int M){
        int S = 0 ;
        for(int i=N ; i > 0; i = i/2){
                for(int j=1 ; j <= M; j= j * 3) S++ ;
        }
}
```

```
int Fibonacci(int N){
        if (N == 0 || N == 1)
            return 1;
        else
            return Fibonacci(N -1) + Fibonacci(N -2);
}
```

```
int M6(int X, int Y){
        if (X > 0)
                return M6(X-1, Y);
        else if (Y > 0)
                return M6(0, Y – 1) +
                                M6(0, Y - 1);
        else
                return 0;
}
```

2. La méthode ***ElementDominant*** peut être améliorée en changeant l'incrémentation ***i***++. Trouver la version améliorée et calculer la nouvelle ***WCTC***.

Exercice n°02 :

Trouver la ***ACTC*** (***A****verage* ***C****ase* ***T****ime* ***C****omplexity*) de la méthode suivante en supposant que les cas du problème sont équiprobables :

```
public boolean existe(Arbre A, int E) {
        if (A == null)
                return false ;
        else if (A.valeur() == E)
                return true ;
        else if (A.valeur() > E)
                return existe(A.fils_gauche(), E) ;
        else
                return existe(A.fils_droit(), E) ;
}
```