POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

INF2010

STRUCT. DE DONN. ET ALGORITH.

Travail pratique 2

Groupe 5

Soumis par :
Mounir Lammali – 2141302
Simon Bachand – 2164037

18 octobre 2022

Partie 2a)

Linear Probing						
	Min	Max	Moyenne	Nb de conflits	Temps d'exécution moyen (en ns)	Temps total (en ns)
Array1	1	2	1.364	7	415	486600
Array2	1	7	1.765	51	313	569200
Array3	1	12	2.083	151	272	737000
Quadratic Probing						
	Min	Max	Moyenne	Nb de conflits	Temps d'exécution moyen (en ns)	Temps total (en ns)
Array1	1	2	1.364	8	400	478300
Array2	1	7	1.714	44	322	590400
Array3	1	7	1.974	125	267	794100
Double Probing						
	Min	Max	Moyenne	Nb de conflits	Temps d'exécution moyen (en ns)	Temps total (en ns)
Array1	1	2	1.25	8	757	483600
Array2	1	5	1.538	56	785	665000
Array3	1	8	1.648	112	583	889600

Observations du point 4: Nous sommes en mesure d'observer que le temps d'exécution moyen par insertion a tendance à diminuer plus la taille du tableau est grande. De plus, les méthodes Linear Probing et Quadratic Probing offrent des résultats similaires en termes de temps d'exécution et de temps total, tandis que le Double Probing devient plus long au fur et à mesure qu'il y a davantage d'éléments.

Observations du point 5: On observe bien que nos résultats expérimentaux concordent avec la théorie puisque cette dernière indique que le temps d'insertion suit une complexité O(N), alors que nos résultats démontrent bien que le temps d'insertion total prend davantage de temps lorsque le tableau de données est plus grand, et moins de temps lorsque le tableau est plus petit.

```java
public static void main(String[] args) {
    long start = System.nanoTime();
    LinearProbing<Double> QuadraticHT = new LinearProbing<>();
      QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
      DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array1, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

```
maps-2141302-2164037 [:Main.main()]
22-10-15 11:01 p.m.   458 ms
> Task :Main.main()
[2, 2, 1, 2, 1, 1, 2, 1, 1, 1, 1]
Min: 1
Max: 2
Moyenne: 486600
Conflits: 7
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
      LinearProbing<Double> QuadraticHT = new LinearProbing<>();
    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
      DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array1, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

maps-2141302-2164037 [:Main.main()]

022-10-15 11:05 p.m. 302 ms

```
> Task :Main.main()
[2, 2, 1, 2, 1, 1, 2, 1, 1, 1, 1]
Min: 1
Max: 2
Moyenne: 478300
Conflits: 8
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
      LinearProbing<Double> QuadraticHT = new LinearProbing<>();
      QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
    DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array1, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

maps-2141302-2164037 [:Main.main()]

22-10-15 11:05 p.m. 447 ms

```
> Task :Main.main()
[2, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1]
Min: 1
Max: 2
Moyenne: 483600
Conflits: 8
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
    LinearProbing<Double> QuadraticHT = new LinearProbing<>();
      QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
      DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array2, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

```
maps-2141302-2164037 [:Main.main()]
22-10-15 11:06 p.m.                426 ms
11:06:49 p.m.: Executing ':Main.main()'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
[1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 5, 1, 1, 3, 2, 2, 1, 1, 2, 1, 1, 1, 3, 1, 1, 2, 1, 1, 1, 7, 3, 2, 4]
Min: 1
Max: 7
Moyenne: 569200
Conflits: 51
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
      LinearProbing<Double> QuadraticHT = new LinearProbing<>();
    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
      DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array2, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

```
naps-2141302-2164037 [:Main.main()]
22-10-15 11:12 p.m.                428 ms
> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
[1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 5, 1, 1, 3, 2, 2, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 7, 3, 2, 4]
Min: 1
Max: 7
Moyenne: 590400
Conflits: 44
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
      LinearProbing<Double> QuadraticHT = new LinearProbing<>();
      QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
    DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array2, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

```
aps-2141302-2164037 [:Main.main()]
2-10-15 11:13 p.m.                309 ms
> Task :compileJava UP-TO-DATE
> Task :processResources NO-SOURCE
> Task :classes UP-TO-DATE

> Task :Main.main()
[1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 5, 1, 1, 1, 3, 2, 2, 1, 1, 2, 1, 1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 4, 1, 3, 1, 3, 1]
Min: 1
Max: 5
Moyenne: 665000
Conflits: 56
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
    LinearProbing<Double> QuadraticHT = new LinearProbing<>();
      QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
      DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array3, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
```

```
aps-2141302-2164037 [:Main.main()]
2-10-15 11:15 p.m.                294 ms
> Task :processResources NO-SOURCE
> Task :classes UP-TO-DATE

> Task :Main.main()
[2, 1, 1, 2, 3, 1, 1, 2, 1, 1, 1, 2, 3, 1, 4, 2, 1, 1, 1, 1, 8, 12, 1, 1, 1, 3, 1, 2, 2, 2, 1, 2, 2, 2, 1, 1, 1, 3, 2, 5, 1, 1, 1, 1, 1, 1, 2, 5, 1, 1,
Min: 1
Max: 12
Moyenne: 737000
Conflits: 151
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
      LinearProbing<Double> QuadraticHT = new LinearProbing<>();
    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
      DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array3, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
}
```

```
ps-2141302-2164037 [:Main.main()]
-10-15 11:16 p.m.  292 ms
> Task :Main.main()
[2, 1, 1, 2, 3, 1, 1, 2, 1, 1, 1, 2, 3, 1, 4, 2, 1, 1, 1, 1, 7, 6, 6, 2, 1, 1, 3, 1, 2, 2, 2, 1, 2, 2, 1, 1, 1, 3, 2, 5, 1, 1, 1, 1, 1, 1, 2, 4,
Min: 1
Max: 7
Moyenne: 794100
Conflits: 125
```

```java
public static void main(String[] args) {
    long start = System.nanoTime();
      LinearProbing<Double> QuadraticHT = new LinearProbing<>();
      QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<>();
    DoubleHashing<Double> QuadraticHT = new DoubleHashing<>();

    fromArrayToHashTable(Array3, QuadraticHT);
    ArrayList<Integer> amas = new ArrayList<>();
    long end = System.nanoTime();
    for (int i = 0; i<QuadraticHT.array.length; i++){
        int count = 0;
        while (i<QuadraticHT.array.length && QuadraticHT.array[i] != null){
            count++;
            i++;
        }
        if (count != 0)
            amas.add(count);
    }
    long tempsTotal = end-start;
    System.out.println(amas);
    System.out.println("Min: " + min(amas));
    System.out.println("Max: " + max(amas));
    System.out.println("Moyenne: " + tempsTotal);
    System.out.println("Conflits: " + QuadraticHT.conflits);
```

```
ps-2141302-2164037 [:Main.main()]
-10-15 11:18 p.m.                301 ms
> Task :Main.main()
[1, 2, 1, 1, 1, 3, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 1, 3, 1, 8, 6, 1, 1, 1, 1, 1
Min: 1
Max: 8
Moyenne: 889600
Conflits: 112
```

Partie 2b)

Linear Probing Nombre de conflits		
Load Factor	0.25	0.75
Array1	3	24
Array2	17	107
Array3	50	402
Quadratic Probing Nombre de conflits		
Load Factor	0.25	0.75
Array1	3	17
Array2	17	89
Array3	47	280
Double Probing Nombre de conflits		
Load Factor	0.25	0.75
Array1	3	10
Array2	21	128
Array3	47	302

Observations: Pour un load Factor moindre, dans ce cas-ci 0.25 contre 0.5, on observe que le nombre de conflits est largement moindre, autant dans de petits tableaux que de plus grands. Quant à un load Factor plus grand, on constate que le nombre de conflits est bien plus grand. Ainsi, lorsqu'on augmente le load Factor, on peut s'attendre à davantage de conflits.

Partie 2c)

	Nb de conflits	Temps d'exécution total (en ns)
Array1	14	460000
Array2	110	500000
Array3	924	480000
	Nb de conflits	Temps d'exécution total (en ns)
Array1	10	450000
Array2	155	530000
Array3	471	490000
	Nb de conflits	Temps d'exécution total (en ns)
Array1	17	480000
Array2	105	520000
Array3	622	515000

Observations: La complexité de l'opération du rehash est O(N) puisqu'il y a N éléments sur lesquels effectuer un rehash, et la taille du tableau double presqu'à 2N puisqu'on double la taille du tableau jusqu'au plus grand nombre premier sous deux fois la taille du tableau.

```java
public static void main(String[] args) {

    LinearProbing<Double> LinearHT = new LinearProbing<Double>();
    fromArrayToHashTable(Array1, LinearHT);

    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<Double>();
    fromArrayToHashTable(Array1, QuadraticHT);

    DoubleHashing<Double> DoubleHT = new DoubleHashing<Double>();
    fromArrayToHashTable(Array1, DoubleHT);

    // Avec Array1 et load Factor est de 0.25 :
    System.out.println(LinearHT.conflits);
    System.out.println(QuadraticHT.conflits);
    System.out.println(DoubleHT.conflits);
}
```

```
-15 6:18 p.m.    317 ms
6:18:41 p.m.: Executing ':Main.main()'...

> Task :compileJava UP-TO-DATE
> Task :processResources NO-SOURCE
> Task :classes UP-TO-DATE

> Task :Main.main()
3
3
3
```

```java
public static void main(String[] args) {

    LinearProbing<Double> LinearHT = new LinearProbing<Double>();
    fromArrayToHashTable(Array1, LinearHT);

    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<Double>();
    fromArrayToHashTable(Array1, QuadraticHT);

    DoubleHashing<Double> DoubleHT = new DoubleHashing<Double>();
    fromArrayToHashTable(Array1, DoubleHT);

    // Avec Array1 et load Factor est de 0.75 :
    System.out.println(LinearHT.conflits);
    System.out.println(QuadraticHT.conflits);
    System.out.println(DoubleHT.conflits);;
}
```

```
0-15 6:28 p.m.  581 ms
6:28:55 p.m.: Executing ':Main.main()'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
24
17
10
```

```java
public static void main(String[] args) {

    LinearProbing<Double> LinearHT = new LinearProbing<Double>();
    fromArrayToHashTable(Array2, LinearHT);

    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<Double>();
    fromArrayToHashTable(Array2, QuadraticHT);

    DoubleHashing<Double> DoubleHT = new DoubleHashing<Double>();
    fromArrayToHashTable(Array2, DoubleHT);

    // Avec Array2 et load Factor est de 0.25 :
    System.out.println(LinearHT.conflits);
    System.out.println(QuadraticHT.conflits);
    System.out.println(DoubleHT.conflits);;
}
```

```
0-15 6:30 p.m.  514 ms
6:30:04 p.m.: Executing ':Main.main()'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
17
17
21
```

```java
public static void main(String[] args) {

    LinearProbing<Double> LinearHT = new LinearProbing<Double>();
    fromArrayToHashTable(Array2, LinearHT);

    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<Double>();
    fromArrayToHashTable(Array2, QuadraticHT);

    DoubleHashing<Double> DoubleHT = new DoubleHashing<Double>();
    fromArrayToHashTable(Array2, DoubleHT);

    // Avec Array2 et load Factor est de 0.75 :
    System.out.println(LinearHT.conflits);
    System.out.println(QuadraticHT.conflits);
    System.out.println(DoubleHT.conflits);;
}
```

```
-15 6:32 p.m.   484 ms
6:32:06 p.m.: Executing ':Main.main()'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
107
89
128
```

```java
public static void main(String[] args) {

    LinearProbing<Double> LinearHT = new LinearProbing<Double>();
    fromArrayToHashTable(Array3, LinearHT);

    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<Double>();
    fromArrayToHashTable(Array3, QuadraticHT);

    DoubleHashing<Double> DoubleHT = new DoubleHashing<Double>();
    fromArrayToHashTable(Array3, DoubleHT);

    // Avec Array3 et load Factor est de 0.25 :
    System.out.println(LinearHT.conflits);
    System.out.println(QuadraticHT.conflits);
    System.out.println(DoubleHT.conflits);;
}
```

```
15 6:33 p.m.   507 ms
6:33:24 p.m.: Executing ':Main.main()'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
50
47
47
```

```java
public static void main(String[] args) {

    LinearProbing<Double> LinearHT = new LinearProbing<Double>();
    fromArrayToHashTable(Array3, LinearHT);

    QuadraticProbing<Double> QuadraticHT = new QuadraticProbing<Double>();
    fromArrayToHashTable(Array3, QuadraticHT);

    DoubleHashing<Double> DoubleHT = new DoubleHashing<Double>();
    fromArrayToHashTable(Array3, DoubleHT);

    // Avec Array3 et load Factor est de 0.75 :
    System.out.println(LinearHT.conflits);
    System.out.println(QuadraticHT.conflits);
    System.out.println(DoubleHT.conflits);;
}
```

```
-15 6:34 p.m.   501 ms
6:34:46 p.m.: Executing ':Main.main()'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :Main.main()
402
280
302
```