Data Structures and Algorithms Laboratory		
Laboratory 8: Hash Tables	**School of Information Technology**	
Name:Sai Hae Naing Lay	**ID:6531501208**	**Section:4**
Date: 2 April 2023	**Due date: on the LMS**	

Objective

- To implement a hash table of String

Exercise 1: (In-class) From the given class diagram, create an array-based hash table and complete the program to get the results as shown.

The objective of this program is to keep pairs of French and English terms in an array **fr** and an array **eng** respectively.

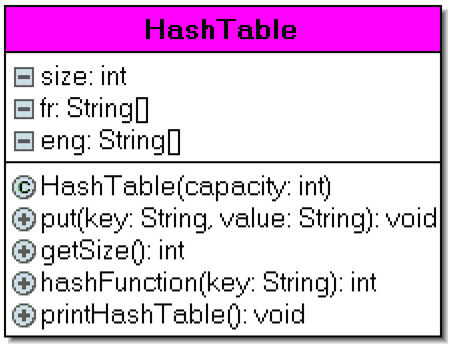
Assume that a hash function is java hashcode (Object.hashCode( )) and the array's size to keep terms is 5.

The input keys and values are:

French	English
pomme	apple
pain	bread
clavier	keyboard

By using Java hash code, no collision happens for these data.

Class diagram:



Expected result:

```
Putting apple is successful.
Putting bread is successful.
Putting keyboard is successful.

--- Hash Table ---
```

```
0 null null
1 clavier keyboard
2 pomme apple
3 null null
4 pain bread
Table size = 3
```

Codes:
Class HashTable

```
package DSA_Lab08;

public class HashTable

{

private int size = 0;

private String[] fr;

private String[] eng;

public HashTable(int capacity) //Constructor method

{

fr = new String[capacity];

eng = new String[capacity];

}

public void put(String key,String value) //add Data to the table

{

//is hashtable full?

if(size == fr.length) //No. of data Equal to array's size or not?

{

System.out.println("Hash Table is full");

return;

}

//use hash function to convert key to hash index

int index = hashFunction(key);

//add new value to the array at the computed position

fr[index] = key;

eng[index] = value;
size++;

System.out.println("Putting " + value + " is sucessful.");

}

public int getSize()//Getter

{

return size;

}

public int hashFunction(String key)//Convert the String key to the
integer key

{

int hashcode = (key.hashCode()&0x7FFFFFFF) % eng.length;

return hashcode;

}

public void printHashTable()

{

System.out.println("--- Hash Table ---");

for(int i = 0; i < fr.length; i++)

{

System.out.println(i + "  " + fr[i] + "  " + eng[i]);

}

}

}
```

Class MainHash

```
public class MainHash {
	public static void main(String[] args) {
		HashTable hashTable = new HashTable(5);
```

```
            String key1 = "pomme";
            String value1 = "apple";
            String key2 = "pain";
            String value2 = "bread";
            String key3 = "clavier";
            String value3 = "keyboard";

            //put keys and values to hashtable
            hashTable.put(key1, value1);
            hashTable.put(key2, value2);
            hashTable.put(key3, value3);
            System.out.println();

            //show data and size of the hashtable
            hashTable.printHashTable();
            System.out.println("Table size = " + hashTable.getSize());
            System.out.println();
      }
}
```

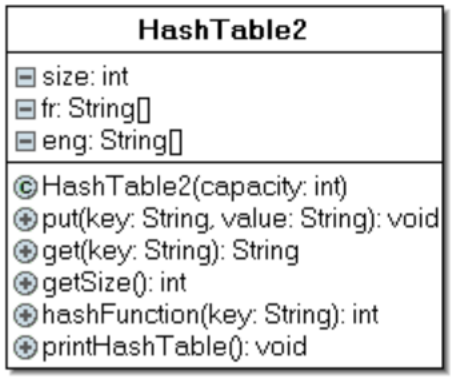
<u>Exercise 2</u>: (Homework) Improve the program in exercise 1 to solve a collision by using a **linear probing technique**. The input keys and values are as follows:

French	English
pomme	apple
pain	bread
clavier	keyboard
ordinateur	computer
lait	milk

Note that there will be collisions of hash codes for these data.

Then search for each key and get its value.

The complete class diagram is:



Expected Output:

```
Putting apple is successful.
Putting bread is successful.
Putting keyboard is successful.
Putting computer has collision!
Putting computer is successful.
Putting milk has collision!
Putting milk is failed!

--- Hash Table ---
0 null null
1 clavier keyboard
2 pomme apple
3 ordinateur computer
4 pain bread
Table size = 4

--- Searching ---
pomme=apple
```

```
pain=bread
clavier=keyboard
ordinateur=computer
lait=N/A
```

Codes:

Class HashTable2

```
package DSA_Lab08;

public class HashTable2

{

private int size = 0;

private String[] fr;

private String[] eng;

public HashTable2(int capacity)

{

fr = new String[capacity];

eng = new String[capacity];

}

public void put(String key,String value)

{

if(size == fr.length)

{

System.out.println("Hash Table is full.");

return;

}

int index = hashFunction(key);

if(fr[index] == null)

{
```

```
fr[index] = key;

eng[index] = value;

size++;

System.out.println("Putting " + value + " is successful.");

}

else if(fr[index] != null)

{

System.out.println("Putting " + value + " has collision!");

while(fr[index] != null && index < size)

{

index++;

}

if(fr[index] == null)

{

fr[index] = key;

eng[index] = value;

size++;

System.out.println("Putting " + value + " is successful.");

}

else

{

System.out.println("Putting " + value + " is failed!");

}

}

}
```

```
public String get(String key)

{

int index = hashFunction(key);

while(fr[index].equals(key) && index < size)

{

index++;

}

if(fr[index] != null && fr[index].equals(key))

{

return eng[index];

}

else

{

return "N/A";

}

}

public int getSize()

{

return size;

}

public int hashFunction(String key)

{

int hashcode = (key.hashCode()&0x7FFFFFFF) % fr.length;

return hashcode;

}
```

```
public void printHashTable()

{

System.out.println("--- Hash Table ---");

for(int i = 0; i < fr.length; i++)

{

System.out.println(i + " " + fr[i] + " " + eng[i]);

}

}

}
```

Class MainHash2

```
public class MainHash2 {
	public static void main(String[] args) {

		HashTable2 hashTable = new HashTable2(5);
		String key1 = "pomme";
		String value1 = "apple";
		String key2 = "pain";
		String value2 = "bread";
		String key3 = "clavier";
		String value3 = "keyboard";
		String key4 = "ordinateur";
		String value4 = "computer";
		String key5 = "lait";
		String value5 = "milk";

		hashTable.put(key1, value1);
		hashTable.put(key2, value2);
		hashTable.put(key3, value3);
		hashTable.put(key4, value4);
		hashTable.put(key5, value5);
		System.out.println();

		hashTable.printHashTable();
		System.out.println("Table size = " + hashTable.getSize());
		System.out.println();

		System.out.println("--- Searching ---");
		System.out.println(key1 + "=" + hashTable.get(key1));
		System.out.println(key2 + "=" + hashTable.get(key2));
		System.out.println(key3 + "=" + hashTable.get(key3));
		System.out.println(key4 + "=" + hashTable.get(key4));
		System.out.println(key5 + "=" + hashTable.get(key5));
```

```
        }
}
```