Ενδεικτικοί κώδικες οδήγησης 7-segment Display με τη χρήση πινάκων ακεραίων αριθμών

```
1 //test 7 segment display
2
3 int pins[] = {2,3,4,5,6,7,8,9};
4
5 void setup ()
6                 {
7                         for(int i=0; i<=7; i++)
8                                 {
9                                         pinMode(pins[i], OUTPUT);
10                                        digitalWrite(i, LOW);
11                                }
12                }
13
14 void loop()
15                {
16                        for(int i=0; i<=7; i++)
17                                {
18                                        digitalWrite(pins[i], HIGH);
19                                        delay( 1000);
20                                        digitalWrite(pins[i], LOW);
21                                }
22                }
```

```
1 // 7-segment-display
2
3 byte digitArray [11][7] =        {{ 1,1,1,1,1,1,0 }, // = 0
4                                   { 0,1,1,0,0,0,0 }, // = 1
5                                   { 1,1,0,1,1,0,1 }, // = 2
6                                   { 1,1,1,1,0,0,1 }, // = 3
7                                   { 0,1,1,0,0,1,1 }, // = 4
8                                   { 1,0,1,1,0,1,1 }, // = 5
9                                   { 1,0,1,1,1,1,1 }, // = 6
10                                  { 1,1,1,0,0,0,0 }, // = 7
11                                  { 1,1,1,1,1,1,1 }, // = 8
12                                  { 1,1,1,0,0,1,1 }, // = 9
13                                  { 0,0,0,0,0,0,0 } // = off
14                                 };
15 int pins[] = {2,3,4,5,6,7,8,9};
16
17 void setup ()
18                 {
19                 for(int i=0; i < 7; i++)
20                         {
21                         pinMode(pins[i], OUTPUT);
22                         }
23                 }
24
25 void loop ()
26                 {
27                 setDigit (10); // display off
28                 delay (500);
29                 for (byte count = 0; count < 10; ++count)
30                         {
31                         setDigit (count);
32                         delay (700);
33                         }
34                 }
35
36 // Καθορισμός συνάρτησης ενός ψηφίου
37 void setDigit (byte digit)
38                 {
39                 for (int i = 0; i < 7; ++i)
40                         {
41                         digitalWrite (pins[i], digitArray[digit][i]);
42                         }
43                 }
```

// πίνακες ακεραίων αριθμών
// ένας πίνακας ακεραίων **n** μεγέθους
// αποθηκεύει **n** ακέραιους αριθμούς
// ο πρώτος στη θέση **[0]**
// ο τελευταίος στη θέση **[n-1]**

```
int array_ints[ ] = {2, 4, 8, 3, 6}; // χωρίς δήλωση μεγέθους
int array_ints[5] = {2, 4, 8, 3, 6}; // με δήλωση μεγέθους (n = 5)
int x = array_ints[1]; // x = 4
```

What is the difference between i++ & ++i in a for loop?

They both increment the number. `++i` is equivalent to `i = i + 1`.
`i++` and `++i` are very similar but not exactly the same. Both increment the number, but `++i` increments the number before the current expression is evaluted, whereas `i++` increments the number after the expression is evaluated.

```
int i = 3;
int a = i++; // a = 3, i = 4
int b = ++a; // b = 4, a = 4
```

The way for loop is processed is as follows
1 First, initialization is performed (i=0)
2 the check is performed (i < n)
3 the code in the loop is executed.
4 the value is incremented
5 Repeat steps 2 - 4
This is the reason why, there is no difference between i++ and ++i in the for loop which has been used.

The difference is that the post-increment operator `i++` returns `i` as it was *before* incrementing, and the pre-increment operator `++i` returns `i` as it is *after* incrementing. If you're asking about a typical `for` loop:

```
for (i = 0; i < 10; i++)
```

or

```
for (i = 0; i < 10; ++i)
```

They're exactly the same, since you're not using `i++` or `++i` as a part of a larger expression.