Laboratory practice No. 2: Big O Notation

Juan S. Cárdenas Rodríguez
Universidad EAFIT
Medellín, Colombia
jscardenar@eafit.edu.co

David Plazas Escudero
Universidad EAFIT
Medellín, Colombia
dplazas@eafit.edu.co

September 5, 2017

1) *ONLINE EXERCISES (CODINGBAT)*

1.a. *Array II*

i.

```
public int[] zeroFront(int[] nums) {            // c0
    boolean [] used = new boolean [nums.length]; // c1
    int cont = 0;                               // c2
    for (int i = 0; i < nums.length; i++) {     // c3 + n
      if(nums[i] == 0) {                        // c4 + n
        if (i != cont) {                        // c5 + n
          nums[i] = nums[cont];                 // c6 + n
          nums[cont] = 0;                       // c7 + n
        }
        cont++;                                 // c8 + n
      }
    }
    return nums;                                // c9
  }
```

Therefore, `zeroFront` is $O(c_0 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8 + c_9 + 6n)$. Applying the sum and product properties, `zeroFont` is $O(n)$.

ii.

```
public int[] notAlone(int[] nums, int val) {    // c0
    for(int i = 1; i < nums.length-1; i++) {    // c1 + n
      if(nums[i] == val && nums[i-1] != val
        && nums[i+1] != val) {                  // c2 + n
        if (nums[i-1] > nums[i+1])              // c3 + n
          nums[i] = nums[i-1];                  // c4 + n
        else                                    // c5 + n
          nums[i] = nums[i+1];                  // c6 + n
        }
      }
      return nums;                                   // c7
    }
```

Therefore, `notAlone` is $O(c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + 6n)$. Applying the sum and product properties, `notAlone` is $O(n)$.

iii.

```
public int[] post4(int[] nums) {                     // c0
    int [] nArray = new int[0];                      // c1
    for(int i = nums.length-1; i >= 0; i--) {        // c2 + n
      if(nums[i] == 4) {                             // c3
        if (i == nums.length-1)                      // c4
          break;                                     // c5
        else {                                       // c6
          nArray = new int[nums.length - i - 1];     // c7
          for (int j = 0; j < nArray.length; j++) {  // c8 + n
            nArray[j] = nums[i + j + 1];             // c9
          }
          break;                                     // c10
        }
      }
    }
    return nArray;
  }
```

iv.

```
public boolean tripleUp(int[] nums) {
  for (int i = 0; i < nums.length - 2; i++) {
    if(nums[i] + 1 == nums[i+1] && nums[i]
     + 2 == nums[i+2]) return true;
  }
  return false;
}
```

v.

```
public int[] tenRun(int[] nums) {
  int tempMult = 0;
  boolean used = false;
  for(int i = 0; i < nums.length; i++) {
    if (nums[i]%10 == 0) {
      used = true;
      tempMult = nums[i];
    }
    if(used)
      nums[i] = tempMult;
  }
  return nums;
}
```

vi.
```
  public int[] shiftLeft(int[] nums) {
    int [] mod = new int[nums.length];
    if (nums.length==1) return nums;
    for (int i=1; i<nums.length; i++) {
      mod[nums.length-1]=nums[0];
      mod[i-1]=nums[i];
    }
    return mod;
  }
```

1.b. Array III

2) SIMULATION OF PROJECT PRESENTATION QUESTIONS

2.a. ArrayMax

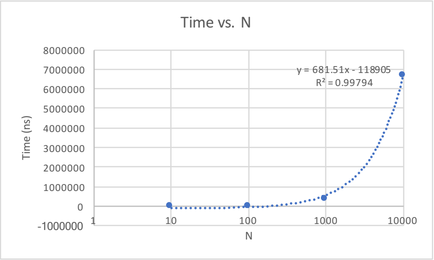

Figure 1: Time vs. N for ArrayMax

N	Time (ns)
10	6000
100	27000
1000	346000
10000	6717000

Table 1: ArrayMax's data.

3) EXAM SIMULATION

i. `start` $+1$, `nums`, `target`

ii. a) $T(n) = T(n/2) + C$

iii. $n - a, a, b, c$
res, solucionar$(n - b, a, b, c)$+1
res, solucionar$(n - c, a, b, c)$+1

iv. e) La suma de los elementos de a y es $O(n)$.

References