Laboratory practice Nro. 2 Complexity of algorithms

Kevin Daniel Torres Parra
Universidad Eafit
Medellín, Colombia
Ktorres2@eafit.edu.co

Vladlen Shatunov
Universidad Eafit
Medellín, Colombia
vshatunov@eafit.edu.co

3) Practice for final project defense presentation

3.1

MergeSort (#Data)	Time (s)	InsertionSort (#Data)	Time (s)
1000	0,00399208	1000	0,029238462
2000	0,00797749	2000	0,149028301
3000	0,0119617	3000	0,307829857
4000	0,01692986	4000	0,541795254
5000	0,02393508	5000	0,810357571
6000	0,02787232	6000	1,291554928
7000	0,03390288	7000	1,708244801
8000	0,03690052	8000	2,645288229
9000	0,04388142	9000	2,965977192
10000	0,04986358	10000	3,395298719
11000	0,0548532	11000	4,069716692
12000	0,05584979	12000	5,235160828
13000	0,06283188	13000	5,702883959
14000	0,06685328	14000	6,753849030
15000	0,0717752	15000	8,385092020
16000	0,07676315	16000	11,513788939
17000	0,08581948	17000	10,542707920
18000	0,09275126	18000	11,484103203
19000	0,10272408	19000	13,399460793
20000	0,10172868	20000	14,015266418

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

3.2

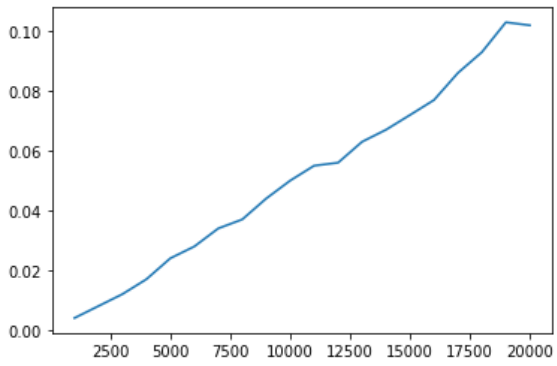

Figure.1(Merge Sort)

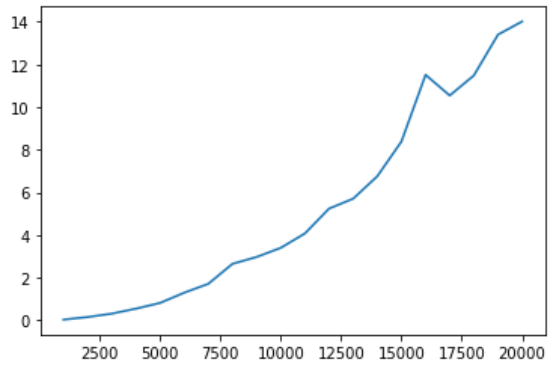

Figure.2 (Insertion Sort)

3.3 - As we can see both in graphics and the table, Insertion Sort takes more time to complete the task, so including it in a video game would be counterproductive, the rendering would be slow which would cost to lower the frequency of frames per second, which would involve low quality graphics and slow motion.

3.4 - Insertion Sort seems to have an asintotic tendency since for the worst case the algorithm would have to go through the whole arrangement of numbers until it finds the least number. On the contrary, Merge Sort has a linear behavior for the worst case, being this the best possible tendency

3.5

Array 2

1.
```
public int countEvens(int[] nums) {
int cont=0;
for(int i=0;i<nums.length;i++){
if(nums[i]%2==0){
cont++;
}
}
return cont;
}
```

2.
```
public int bigDiff(int[] nums) {
 int minimo = nums[0];
 int maximo = nums[0];
 for (int i = 0; i < nums.length; i++){
   minimo = Math.minimo(min,nums[i]);
   maximo = Math.maximo(max,nums[i]);
 }
 return maximo-minimo;
```

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

```
}
```

3.
```
public int sum13(int[] nums) {
int sum = 0;
for (int i=0; i<nums.length; i++) {
if (nums[i]==13) i++;
else sum+=nums[i];
}
return sum;
}
```

4.
```
public boolean isEverywhere(int[] nums, int val) {
for(int i=0;i<nums.length-1;i++){
if(nums[i]!=val&&nums[i+1]!=val)
return false;
}
return true;
}
```

5.
```
public boolean only14(int[] nums) {
for (int i=0; i<nums.length; i++) {
if(nums[i] != 4 && nums[i] != 1) {
return false;
}
}
return true;
}
```

Array3

1)
```
public int maxSpan(int[] nums) {
  int span = 0;
  int x = 0;
  for (int i = 0; i<nums.length;i++){
    for (int j= 0; j<nums.length; j++){
      if (nums[i] == nums[j]){
      x = j-i+1;
      span = Math.max(x,span);
      }
    }
  }
  return span;
}
```

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

2)
```
public int[] seriesUp(int n) {
  int[] array = new int[n*(n+1)/2];
  int limite = 1;
  for (int i = 0; i<n ; i++){
    for (int j = 0; j<=i; j++){
      array[i*(i+1)/2+j] = j+1;
    }
  }
  return array;
}
```

3)
```
public int countClumps(int[] nums) {
  int cont = 0;
for(int i = 0; i<nums.length-1; i++){
if(nums[i] == nums[i+1]){
if(i==0||nums[i-1]!=nums[i]){
cont++;
}
}
}
return cont;
}
```

4)
```
public boolean canBalance(int[] nums) {
  int der = 0;
  int izq = 0;
  for (int i = 0; i<nums.length;i++){
    der = der+nums[i];
  }
  for (int i = 0; i < nums.length;i++){
    izq = izq+nums[i];
    der = der-nums[i];
    if (der == izq){
      return true;
    }
  }
  return false;
}
```

5)
```
public int[] squareUp(int n) {
  int array[] = new int [n*n];
  for (int i = 1; i<=n; i++){
    for (int j = 1; j<=i; j++){
```

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

```
      array[i*n-j]= j;
    }
   }
   return array;
  }
```

3.6 Array2

1)

T2(n) = c1
T3(n) = c2*n +c3
T4(n) = (c4)*n

T(n) = c1 + c3 + c2*n + c4*n
T(n) = n

2)

T2(n) = c1
T3(n) = c2
T4(n) = c3*n + c4
T5(n) = c6*n
T6(n) = c7*n

T(n) = c1 + c2 + c4 + c3*n + c6*n + c7*n
T(n) = n

3)

T2(n) = c1
T3(n) = c2*n + c3
T4(n) = c4*n
T5(n) = c5

T(n) = c1 +c3 + c5 + c2*n + c4*n
T(n) = n

4)

T1(n) = c1*n + c2
T2(n) = c3*n

T(n) = c2 + c1*n + c3*n
T(n) = n

5)

T2(n) = c1*n + c2
T3(n) = c3*n

T(n) = c2 + c1*n + c3*n
T(n) = n

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

Array3

1)

T2(n) = c1
T3(n) = c2
T4(n) = c3*n + c4
T5(n) = c5*n^2 + c6*n
T6(n) = c7*n^2
T7(n) = c8*n^2

T(n) = c1 + c2 + c4 + c6*n + c3*n^2 + c5*n^2 + c7*n^2 + c8*n^2
T(n) = n^2

2)

T2(n) = c1
T3(n) = c2
T4(n) = c3*n + c4
T5(n) = c5*n^2 + c6*n
T6(n) = c7 *n^2

T(n) = c1 + c2 + c4 + c6*n + c3*n^2 + c5*n^2 + c7*n^2
T(n) = n^2

3)

T2(n) = c1
T3(n) = c2*n + c3
T4(n) = c4*n
T5(n) = c5*n

T(n) = c1 + c3 + c2*n + c4*n + c5*n
T(n) = n

4)

T2(n) = c1
T3(n) = c2
T4(n) = c3*n + c4
T5(n) = c5*n
T6(n) = c6*n + c7
T7(n) = c8*n
T8(n) = c9*n
T9(n) = c10*n

T(n) = c1 + c2 + c4 + c7 + c3*n + c5*n + c6*n + c8*n + c9*n + c10*n
T(n) = n

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

5)

T2(n) = c1
T3(n) = c2*n + c3
T4(n) = c4*n^2 + c5*n
T5(n) = c6*n^2

T(n) = c1 + c3 + c2*n + c5*n + c4*n^2 + c6*n^2
T(n) = n^2

3.6 "n" and "m" are variables that shows us how many processes are going to be made during the algorithm we have in mind/ are working on.

4) Practice for midterms

4.1 c
4.2 b
4.3 b
4.4 b
4.5 d
4.6 a
4.7
4.8 a
4.9 d
4.10 c
4.11 c
4.12 b
4.13 c
4.14 a

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473