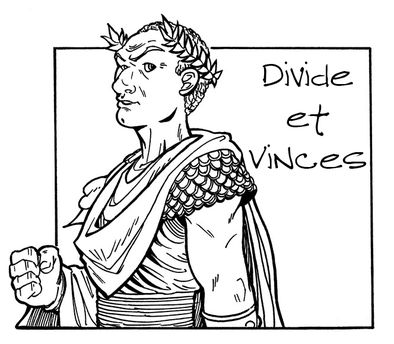
Pablo Mariano Moreno Mancebo

Algorítmica Grupo D - 2

Práctica 2

**Ejercicio 7.**

Dado un vector v[1..n] de valores numéricos enteros, se dice que los elementos v[i] y v[j] realizan una inversión si i<j pero v[i]>v[j]. Proporcionar un algoritmo que devuelva

todas las posibles inversiones del vector de entrada v.

Índice:

[*Caso base:*](#_xv831tqpggqh)

[*Pseudocódigo:*](#_kak7o97q2rs)

[*Implementación:*](#_4vkto0ldvqor)

[*Eficiencia: O ( n )*](#_mklc6mqqytn8)

[*Ejemplo de entrada y salidas esperadas:*](#_1xfkquyqf9l2)

[*Requisitos Divide y Vencerás (Resoluble en DyV)*](#_ypn8u0ygjfd4)

[*Algoritmo DyV*](#_5uf7mv80zod7)

[*Implementación:*](#_a3dc3w68qsa3)

[*Eficiencia: O(log(n))*](#_rzcctbm6h8vw)

[*- Ejemplo de entrada y salidas esperadas*](#_c6ioa2nbyd7q)

[*Análisis Umbral*](#_ahy9muwif7o4)

# Caso base:

## Pseudocódigo:

| *var inversiones*:=0;    vector sol;  Desde i := 0 hasta n -1  Si (v[i]>v[i+1])  inversiones++;  sol.push\_back(i,i+1)  Fin Si  Fin Desde |
| --- |

## 

## Implementación:

| *int* inversiones=0;  *//Para la solución voy a hacer un vector de pairs*  vector<pair<*int*,*int*>> sol;  for(*int* i = 0 ; i < n -1 ; i++)  {  if (v[i]>v[i+1])  {  inversiones++;  sol.push\_back(make\_pair(i,i+1));  }  } |
| --- |

## 

## Eficiencia: O ( n )

| *int* inversiones=0;  *//Para la solucion voy a hacer un vector de pairs*  vector<pair<*int*,*int*>> sol;  for(*int* i = 0 ; i < n -1 ; i++)  {  if (v[i]>v[i+1])  {  inversiones++;  sol.push\_back(make\_pair(i,i+1));  }  } | O(1)  O(1)  O(n)  O(1)  O(1)  O(1) |
| --- | --- |

# 

## Ejemplo de entrada y salidas esperadas:

## 

# Requisitos Divide y Vencerás (Resoluble en DyV)

* El problema se puede dividir en problemas de tamaño menor de forma independiente
* Las soluciones van a seguir siendo de las misma naturaleza (vectores) de manera que se pueden combinar entre sí
* El umbral usaré el caso del vector de tamaño 1

# 

# 

# Algoritmo DyV

## Implementación:

| *void* ObtenerInversiones(*const* vector<*int*> *&v*, list<pair<*int*, *int*>> *&inversiones*, *int* *inicio*, *int* *fin*)  {  *// Tu código aquí*  if(fin - inicio + 1 == 1)*//Umbral*  {    }  else  {  *//Dividir*  *int* medio = ((inicio + fin )/2);  ObtenerInversiones(v,inversiones,inicio,medio);  ObtenerInversiones(v,inversiones,medio+1,fin);  *//Combinar*  if (v[medio] > v[medio+1])  {  inversiones.push\_back(make\_pair(medio,medio+1));  }    }    } |
| --- |

## 

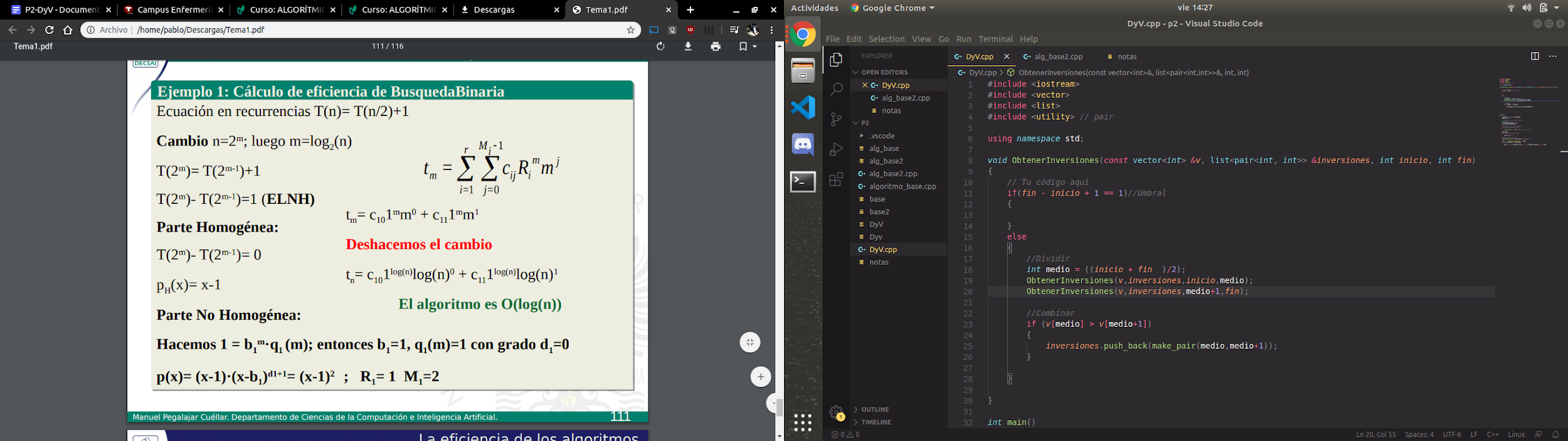
## Eficiencia: O(log(n))

Caso base = O (1 ) por el if(fin - inicio + 1 == 1)//Umbral

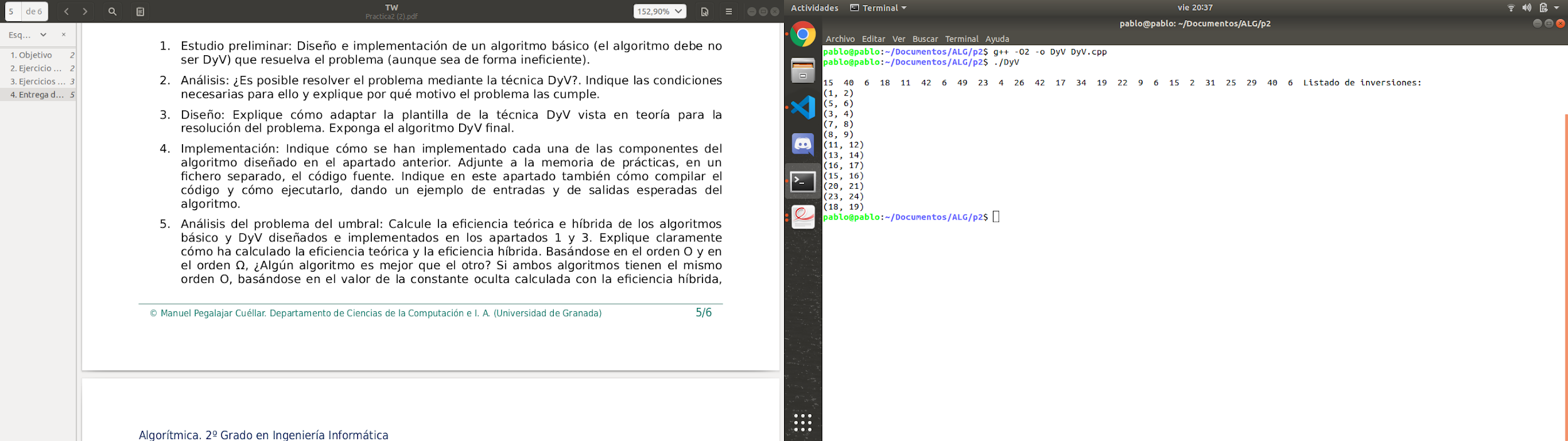
Después dos llamadas recurrentes

* Desde inicio hasta medio → tamaño de ese problema es la mitad
* Desde medio hasta fin la otra mitad

Ecuación de recurrencia T(n) =/(n/2)+1



## - Ejemplo de entrada y salidas esperadas



Con una semilla de = 1234 se generan siempre los mismos resultados en el aleatorio y se generan 25 números

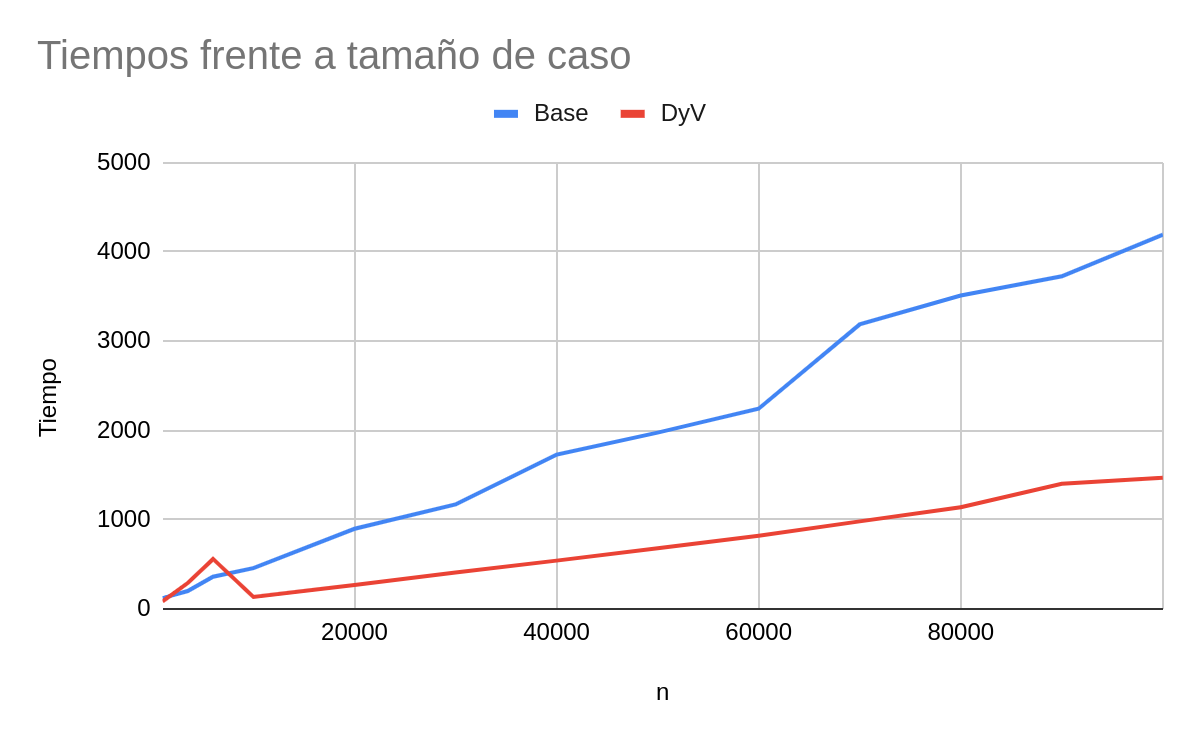
Nota: Pueden salir en orden diferente respecto el algoritmo básico

# Análisis Umbral

Tenemos que ver cuando si Función S = DyV(P,n) , cuando esa P es suficientemente pequeña para aplicar un algoritmo base.

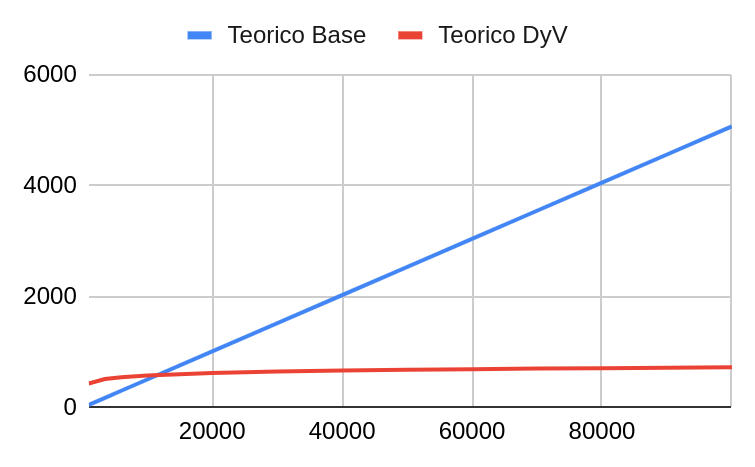
Para eso voy a comparar los tiempos de ejecución del algoritmo DyV y el algoritmo básico

| **Básico** | | **DyV** | |
| --- | --- | --- | --- |
| **n** | **t** | **n** | **t** |
| 1000 | 120 | 1000 | 85 |
| 3500 | 200 | 3500 | 290 |
| 6000 | 360 | 6000 | 558 |
| 10000 | 456,3333333 | 10000 | 134,8333333 |
| 20000 | 897,5 | 20000 | 267,5 |
| 30000 | 1170 | 30000 | 407,1666667 |
| 40000 | 1727,833333 | 40000 | 540,6666667 |
| 50000 | 1974,666667 | 50000 | 680 |
| 60000 | 2242,5 | 60000 | 818,6666667 |
| 70000 | 3186,833333 | 70000 | 980,6666667 |
| 80000 | 3510,333333 | 80000 | 1139,5 |
| 90000 | 3726,333333 | 90000 | 1402,5 |
| 100000 | 4193,166667 | 100000 | 1469 |



Con los datos anteriores calculo k:

| k- base | 0,05072739113 | k - DyV | 43,76348052 |
| --- | --- | --- | --- |

De manera que:

Como observamos en la gráfica y en los datos el punto umbral ronda cerca de un tamaño de caso de 500

Con los datos:

| Teorico Base | Teorico DyV |
| --- | --- |
| 50,72739113 | 436,1374064 |
| 177,5458689 | 515,2335483 |
| 304,3643468 | 549,2643624 |
| 507,2739113 | 581,5165418 |
| 1014,547823 | 625,2800223 |
| 1521,821734 | 650,8800173 |
| 2029,095645 | 669,0435028 |
| 2536,369556 | 683,1321968 |
| 3043,643468 | 694,6434978 |
| 3550,917379 | 704,3761642 |
| 4058,19129 | 712,8069834 |
| 4565,465201 | 720,2434928 |
| 5072,739113 | 726,8956773 |