Algorítmica – Práctica 2 – Divide y Vencerás

David Infante Casas – 75165296P - 2ºD

Ejercicio 1:

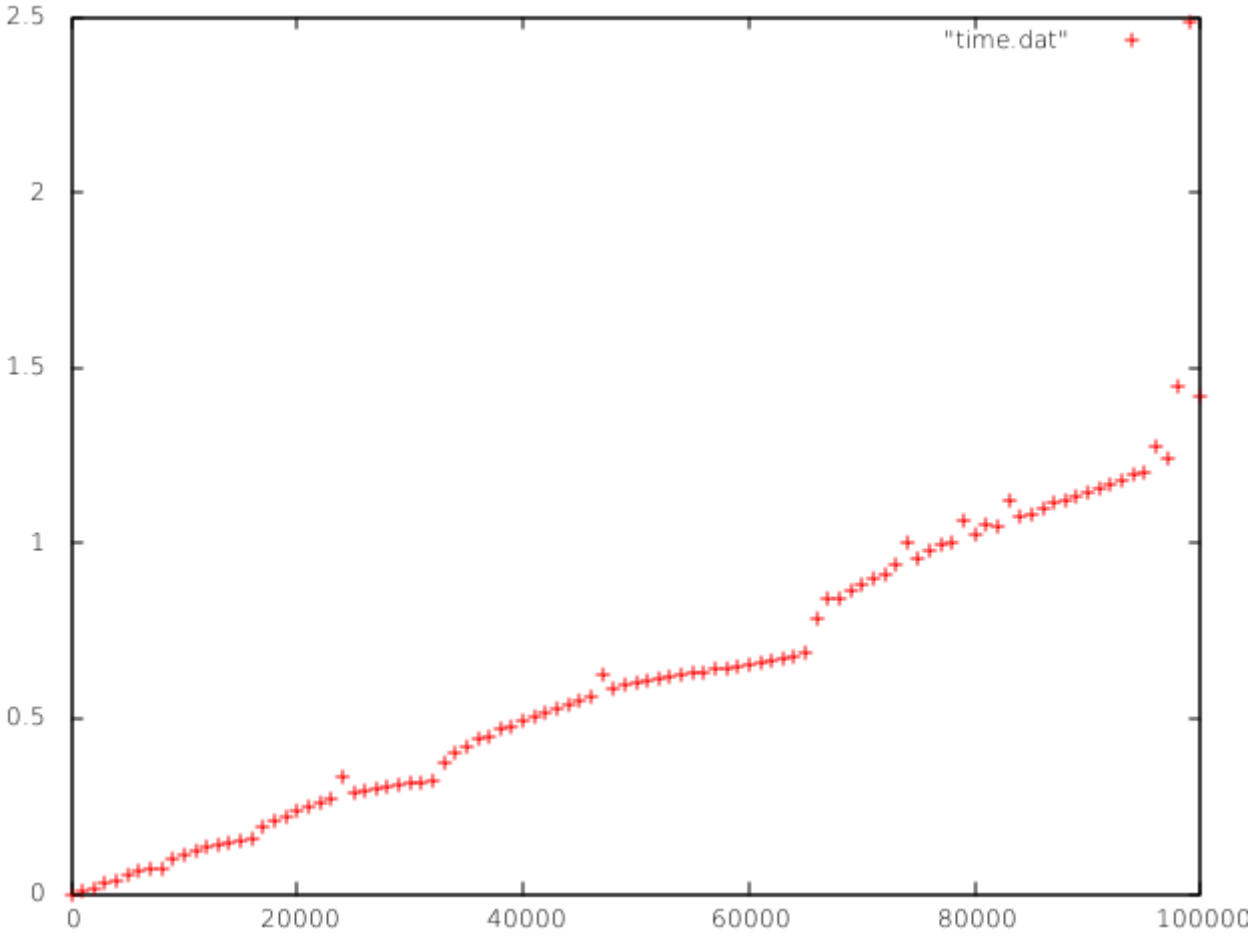
Código:

```
pair<int ,int> Max_Min(const vector<int> & v){

	pair<int, int> max_min;
	int tamanio = v.size();
	int mitad = v.size() / 2;

	if (tamanio == 1) {
		max_min.first = v[0];
		max_min.second = v[0];
	}else if (tamanio == 2) {
		if (v[0] < v[1]) {
			max_min.first = v[0];
			max_min.second = v[1];
		} else if (v[0] >= v[1]){
			max_min.first = v[1];
			max_min.second = v[0];
		}
	}else if (tamanio > 2){
		vector<int> izq, dcha;
		pair<vector<int>, vector<int> > aux;
		for (int i = 0; i < mitad; ++i)
			izq.push_back(v[i]);
		for (int j = mitad; j < tamanio; ++j)
			dcha.push_back(v[j]);
		aux.first = izq;
		aux.second = dcha;

		pair<int, int> p_izq = Max_Min(aux.first);
		pair<int, int> p_dcha = Max_Min(aux.second);

		if (p_izq.first < p_dcha.first)
			max_min.first = p_izq.first;
		else if (p_izq.first >= p_dcha.first)
			max_min.first = p_dcha.first;

		if (p_izq.second < p_dcha.second)
			max_min.second = p_dcha.second;
		else if (p_izq.second >= p_dcha.second)
			max_min.second = p_izq.second;
	}
	return max_min;
}
```

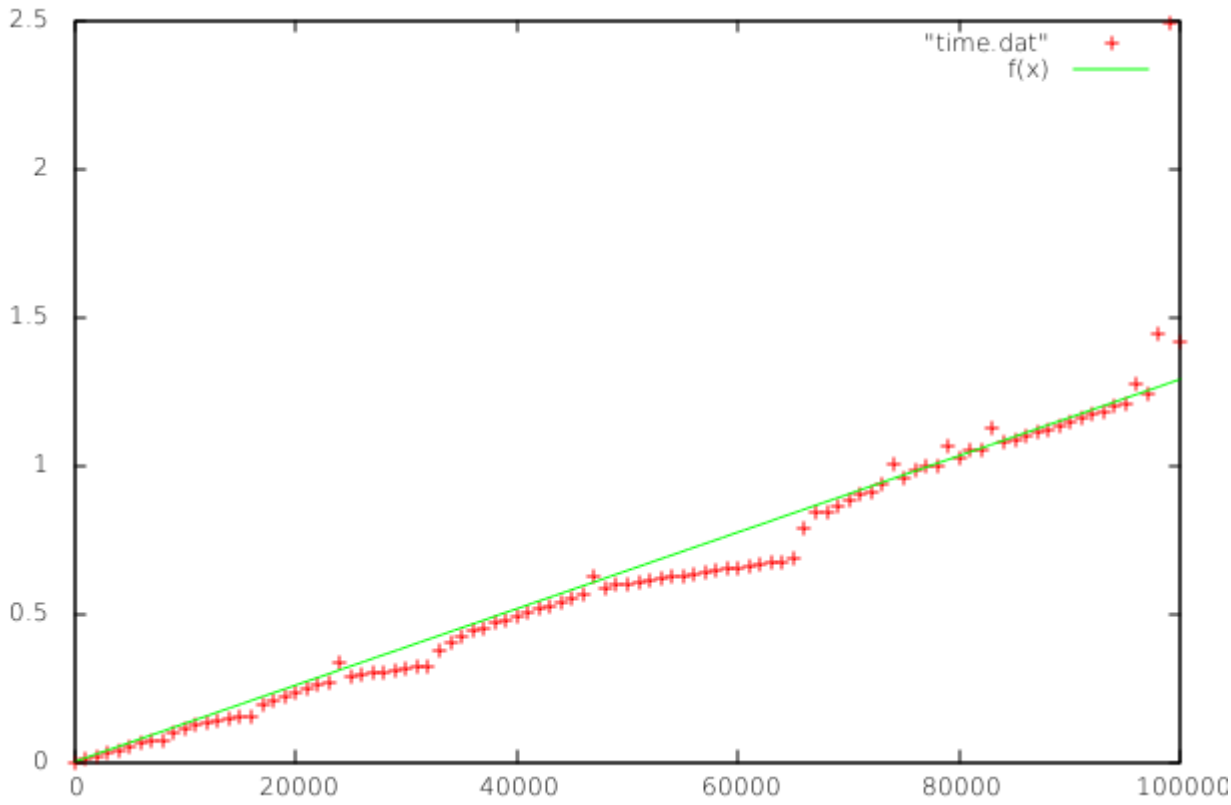
Eficiencia teórica:

O(n*log(n))

Eficiencia empírica:



Ajuste:



Ejercicio 2:

Código:

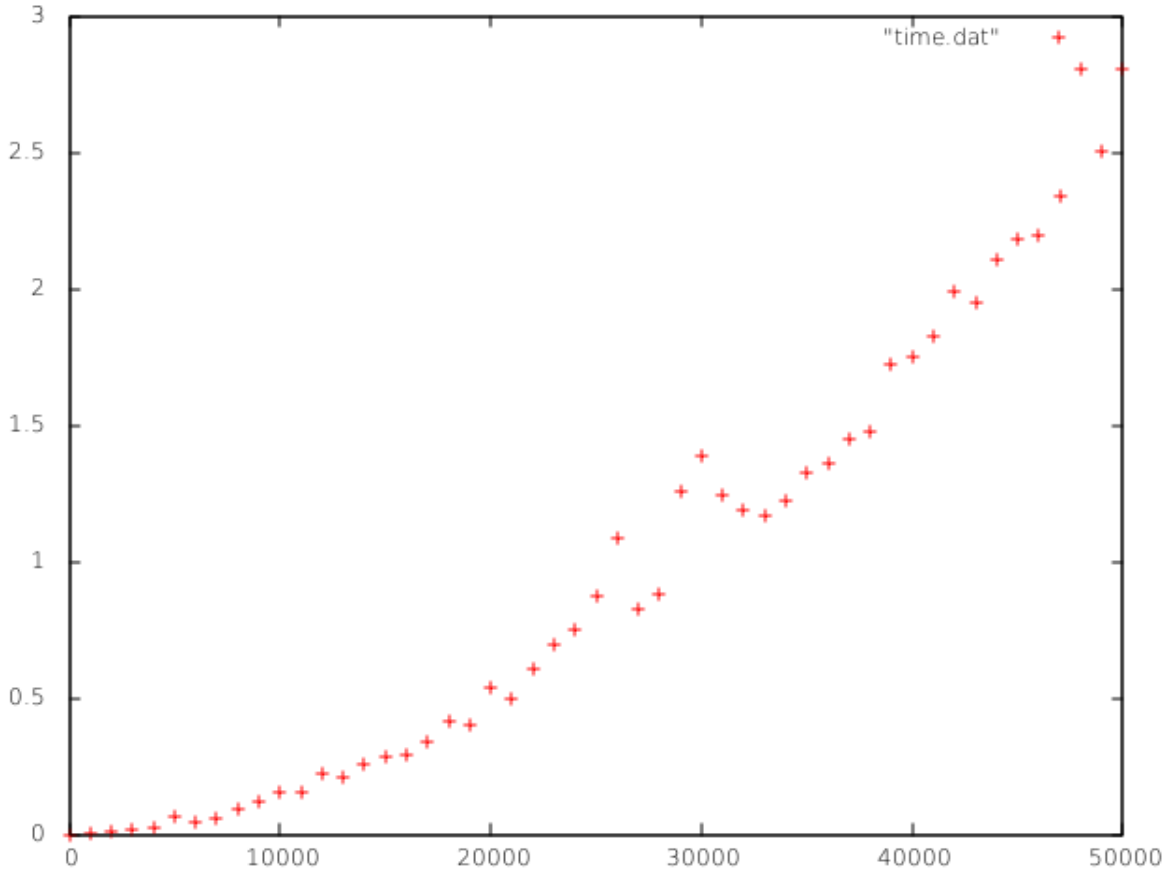
```
bool ordenado = false;

pair<int, int> Moda(int arr[], int left, int right) {
        pair<int, int> moda;
        int *ho=new int[50001];
        int *he=new int[50001];
        int util_ho = 0, util_he = 0;
        int k = 0;
        moda.first = 0;
        moda.second = 0;

        if (!ordenado){
                quickSort(arr, left, right);
                ordenado = true;
        }

        do{
                ho[util_ho] = arr[k];
                ++k;
                ++util_ho;
        }while (arr[k] == arr[k-1]);

        while (util_he < (right – util_ho)){
                he[util_he] = arr[k];
```

```
                ++k;
                ++util_he;
            }

            if (util_ho > util_he && util_ho > 1){
                moda.first = ho[0];
                moda.second = util_ho;
            }
             else{
                pair<int, int> m_he =  Moda(he, util_ho, util_he);
                if (m_he.second > moda.second){
                    moda.first = m_he.first;
                    moda.second = m_he.second;
                }
        }
        return moda;
}
```

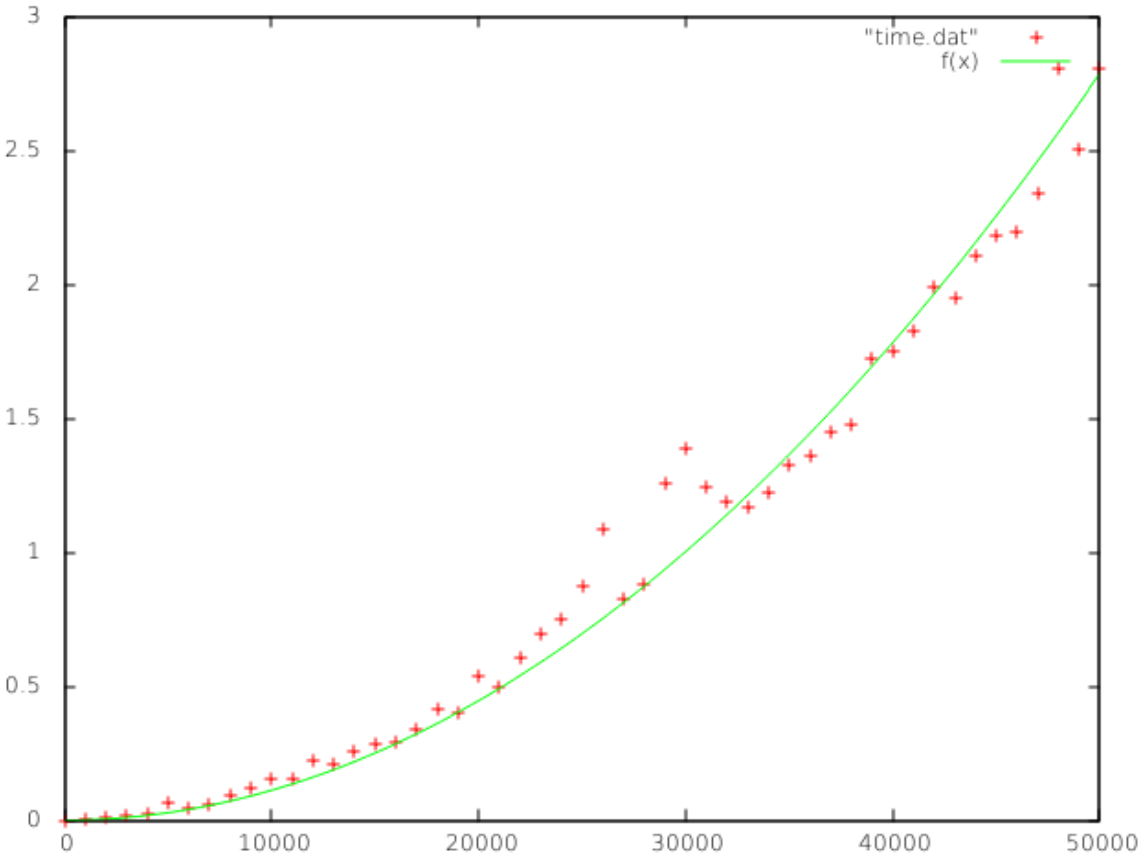
Eficiencia teórica:

$O(n^2)$

Eficiencia empírica:



Ajuste:



Ejercicio 3:

Código:

```
int binarySearch(const vector<int> v, int inicio, int fin, int x){
	if (v.begin() <= v.end()){
		int mitad = v.size()/2;
		if(v[mitad]==x) return mitad;
		else{
			if (x > v[mitad]) return binarySearch(v, mitad+1, v.size()-1, x);
			else return binarySearch(v, inicio, mitad-1, x);
		}
	}else return -1;
}

vector<int> TT(vector<int> tuercas, vector<int> tornillos){
	vector<int> tornillos_tuercas;
	for (int i = 0; i < tuercas.size(); ++i){
		int a = binarySearch(tornillos, 0, tornillos.size(), tuercas.at(i));
		tornillos.push_back(a);
	}
	return tornillos_tuercas;
}
```