



UNA COMPARACIÓN ENTRE C++, GO Y PYTHON

A vibrant space-themed background. The top left features a large planet with horizontal stripes and a smaller ringed planet below it. The bottom right shows a cratered moon. An astronaut in a white suit floats in the upper right, holding a long, looping tether. The background is a deep blue-purple gradient filled with numerous white stars of varying sizes and soft, glowing nebulae in shades of purple and blue.

INTRODUCCIÓN



Go es un lenguaje compilado basado en C y con la facilidad de Python.

C++ es un lenguaje compilado, extensión de C.

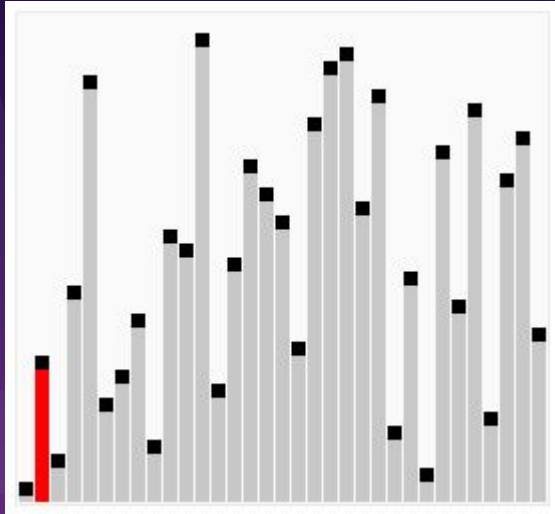
Python es un lenguaje interpretado, con uso científico.



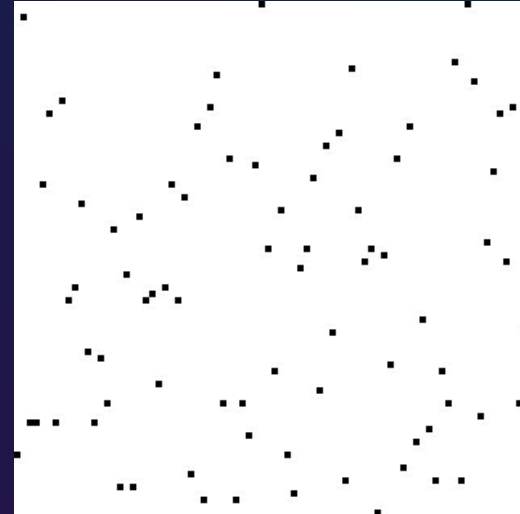


Se pondrá a prueba los 3 lenguajes mencionados
con los algoritmos de ordenamiento:

Cocktail Sort



Counting Sort



ALGORITMOS





COCKTAIL SORT

1. ✦ El primer paso es recorrer la lista de datos de izquierda a derecha, se comparan los valores adyacentes, y si el valor de la izquierda es mayor, se intercambian los valores. El objetivo es poner el valor máximo al final.
2. El segundo paso es recorrer la lista de datos de derecha a izquierda, se comparan los valores adyacentes, y si el valor de la derecha es menor, se intercambian los valores. El objetivo es poner el valor mínimo al inicio.

COUNTING SORT

1. Busca el elemento máximo (max) y el mínimo (min).
2. Crea una lista auxiliar de tamaño $\text{max} + 1$.
3. Se subdivide en los siguientes pasos:
 - Se almacena el recuento de cada elemento de la lista en su índice correspondiente en la lista auxiliar.
 - Se suman los recuentos ($a[i] + a[i-1]$), esto para colocar los elementos en el índice correcto de la lista ordenada.
4. Posiciona los elementos en la lista original o una de salida, para esto se utilizan los índices.

COCKTAIL SORT

Caso	Complejidad
Mejor caso	$O(n)$
Caso promedio	$O(n^2)$
Peor caso	$O(n^2)$

COUNTING SORT

Caso	Complejidad
Mejor caso	$O(n+k)$
Caso promedio	$O(n+k)$
Peor caso	$O(n+k)$


```

vector<int> cocktailSort(vector<int> list) {
    int last = list.size() - 1;
    while(true) {
        bool swapped = false;
        for (int i = 0; i < last; i++) {
            if (list[i] > list[i+1]) {
                int a = list[i];
                list[i] = list[i+1];
                list[i+1] = a;
                swapped = true;
            }
        }
        if(!swapped) return list;

        swapped = false;
        for (int i = last - 1; i >= 0; i--) {
            if (list[i] > list[i+1]) {
                int a = list[i];
                list[i] = list[i+1];
                list[i+1] = a;
                swapped = true;
            }
        }
        if (!swapped) return list;
    }
    return list;
}

```

```

vector<int> CountingSort(vector<int> list) {
    int minItem = *min_element(list.begin(), list.end());
    vector<int> countList ((*max_element(list.begin(), list.end()) - minItem + 1), 0);
    vector<int> outputList (list.size(),0);

    for(int i = 0; i < list.size(); i++) countList[list[i] - minItem] += 1;

    for(int i = 1; i < countList.size(); i++) countList[i] += countList[i-1];

    for (int i = list.size() - 1; i >= 0; i--) {
        outputList[countList[list[i] - minItem] - 1] = list[i];
        countList[list[i] - minItem] -= 1;
    }

    for(int i = 0; i < list.size(); i++) list[i] = outputList[i];

    return list;
}

```

```

+ func cocktailSort(list []int) {
    last := len(list) - 1
    for {
        swapped := false
        for i := 0; i < last; i++ {
            if list[i] > list[i+1] {
                list[i], list[i+1] = list[i+1], list[i]
                swapped = true
            }
        }
        if !swapped {
            return
        }
        swapped = false
        for i := last - 1; i >= 0; i-- {
            if list[i] > list[i+1] {
                list[i], list[i+1] = list[i+1], list[i]
                swapped = true
            }
        }
        if !swapped {
            return
        }
    }
}

```

```

+ func countingSort(list []int) {
    minItem := Min(list)
    countList := Zero(Max(list) - minItem + 1)
    outputList := Zero(len(list))

    for i := 0; i < len(list); i++ {
        countList[list[i]-minItem] += 1
    }

    for i := 1; i < len(countList); i++ {
        countList[i] += countList[i-1]
    }

    for i := len(list) - 1; i >= 0; i-- {
        outputList[countList[list[i]-minItem]-1] = list[i]
        countList[list[i]-minItem] -= 1
    }

    for i := 0; i < len(list); i++ {
        list[i] = outputList[i]
    }
}

```

```

def cocktailSort(list):
    last = len(list) - 1
    while True:
        swapped = False
        for i in range(last-1,1,-1):
            if list[i] > list[i+1]:
                list[i], list[i+1] = list[i+1], list[i]
                swapped = True

        if swapped == False: return

        swapped = False
        for i in range(last-1,-1,-1):
            if list[i] > list[i+1]:
                list[i], list[i+1] = list[i+1], list[i]
                swapped = True

        if swapped == False: return

```

```

def CountingSort(list):
    minItem = min(list)
    countList = [0] * (max(list) - minItem + 1)
    outputList = [0] * len(list)

    for i in range(0, len(list)): countList[list[i] - minItem] += 1

    for i in range(1, len(countList)): countList[i] += countList[i-1]

    for i in range(len(list)-1, -1, -1):
        outputList[countList[list[i] - minItem] - 1] = list[i]
        countList[list[i] - minItem] -= 1

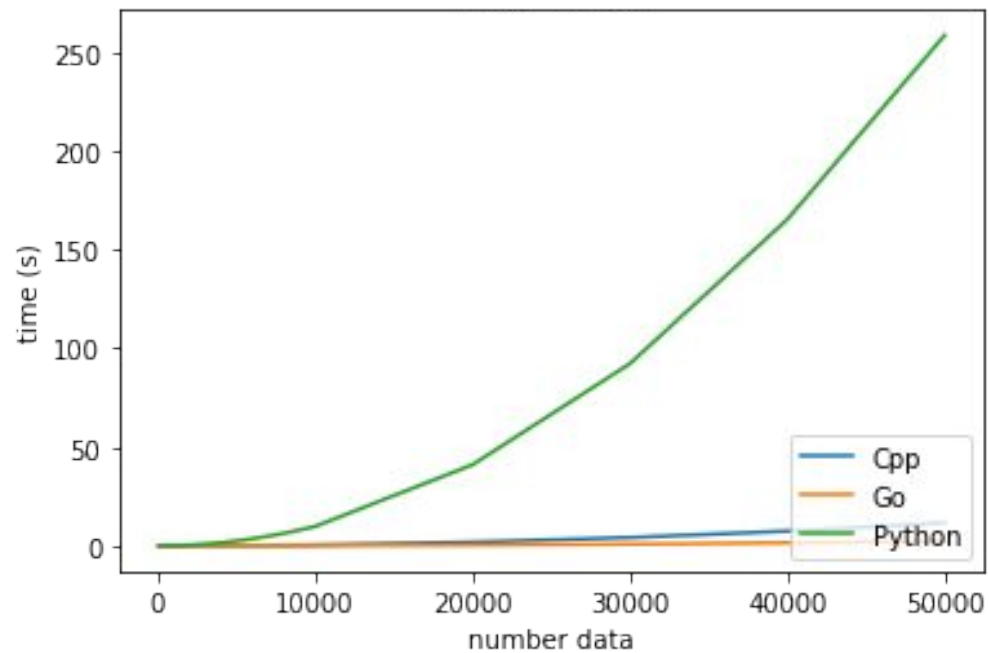
    for i in range(0, len(list)): list[i] = outputList[i]

```

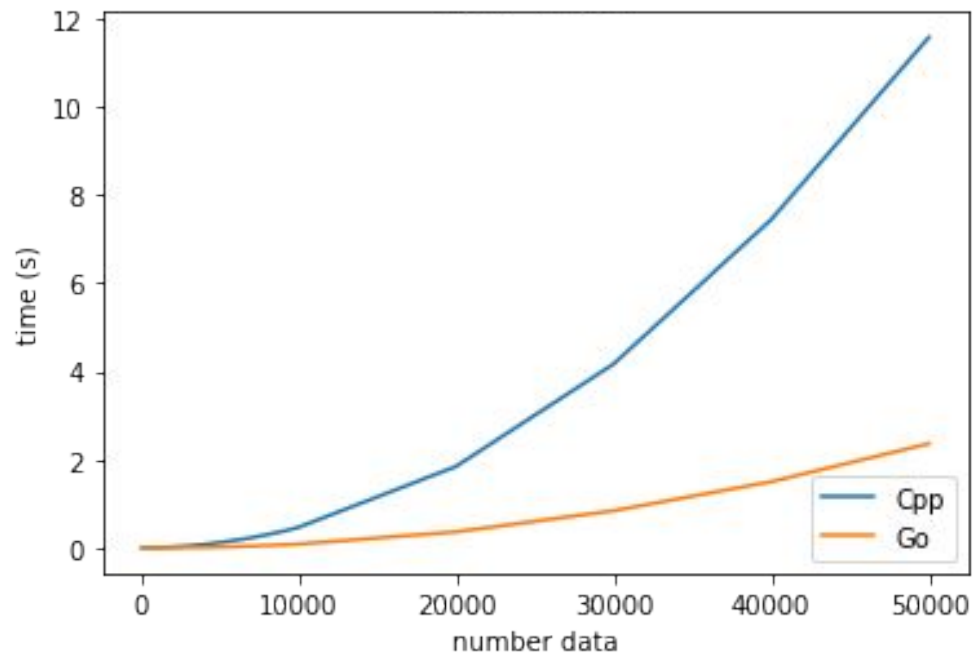
The background is a deep space scene. In the top left, there's a large planet with horizontal stripes. Below it is a smaller planet with a ring. In the top right, an astronaut in a white suit is floating, holding a long, thin, looping rope. In the bottom right, there's a large, cratered moon. The entire scene is filled with numerous small white stars and larger, four-pointed starburst shapes. The color palette transitions from dark blue at the top to a rich purple at the bottom.

COMPARACIÓN

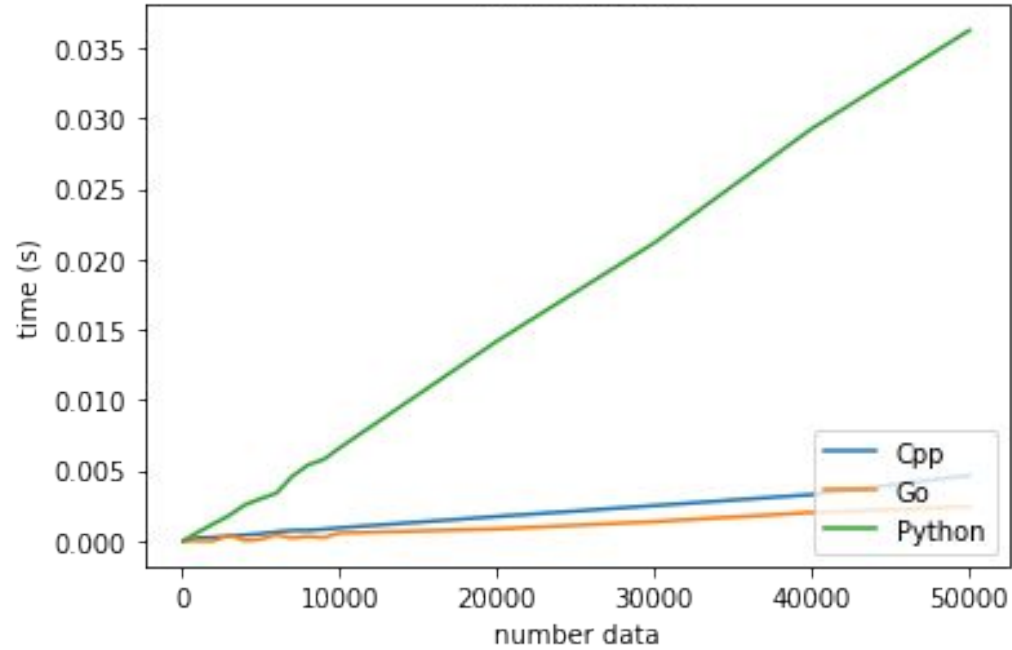
Cocktail Sort



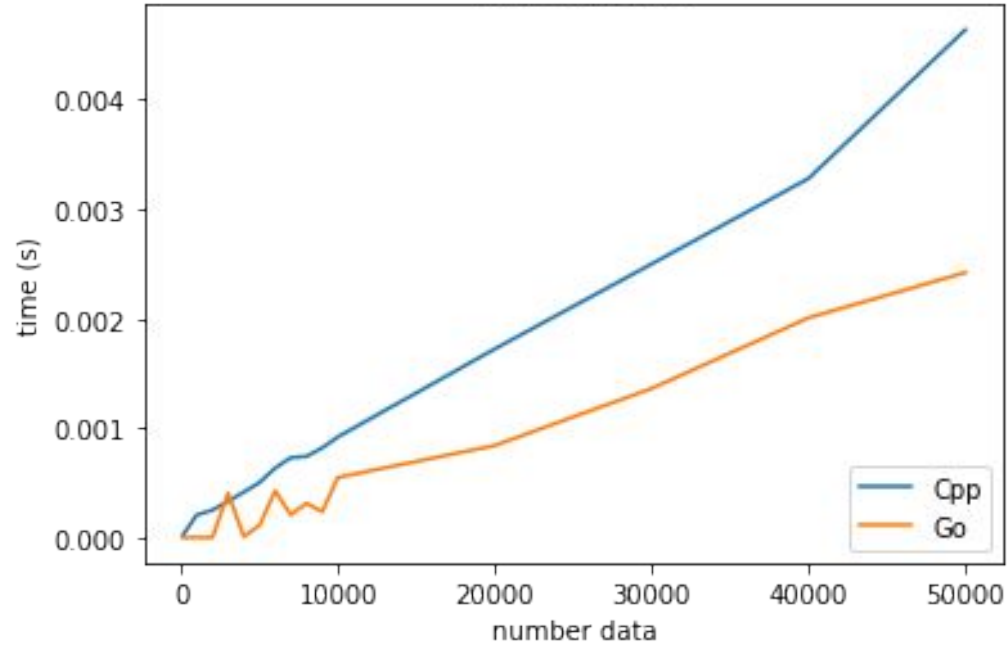
Cocktail Sort



Counting Sort



Counting Sort



Counting Sort Go

N	1	2	3	4	5	AVG	SD
100	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
1000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
2000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
3000	0.00100780	0.00050640	0.00000000	0.00000000	0.00050650	0.00040414	0.000377359
4000	0.00004710	0.00000000	0.00000000	0.00000000	0.00000000	0.00000942	0.000018840
5000	0.00000000	0.00000000	0.00000000	0.00054790	0.00000000	0.00010958	0.000219160
6000	0.00053600	0.00107970	0.00051180	0.00000000	0.00000000	0.00042550	0.000402427
7000	0.00051750	0.00000000	0.00000000	0.00000000	0.00051670	0.00020684	0.000253326
8000	0.00000000	0.00053940	0.00050740	0.00000000	0.00051680	0.00031272	0.000255547
9000	0.00000000	0.00000000	0.00062060	0.00055750	0.00000000	0.00023562	0.000289263
10000	0.00051380	0.00055530	0.00013060	0.00050420	0.00104120	0.00054902	0.000290036
20000	0.00051370	0.00108970	0.00101970	0.00099720	0.00058210	0.00084048	0.000241800
30000	0.00158390	0.00161050	0.00157420	0.00099800	0.00104110	0.00136154	0.000279819
40000	0.00216950	0.00223260	0.00153590	0.00199770	0.00210200	0.00200754	0.000248333
50000	0.00322110	0.00267620	0.00206540	0.00199480	0.00214750	0.00242100	0.000466516

CONCLUSIÓN

The background is a deep purple and blue space scene. In the top left, there's a large planet with horizontal stripes. Below it is a smaller planet with a ring. In the top right, an astronaut in a white suit is floating, holding a long, thin, looping rope. In the bottom right, there's a large, cratered moon. The entire scene is filled with numerous small white stars and larger, four-pointed starburst shapes. There are also some soft, wavy, light-colored shapes in the background, possibly representing nebulae or gas clouds.

FIN

