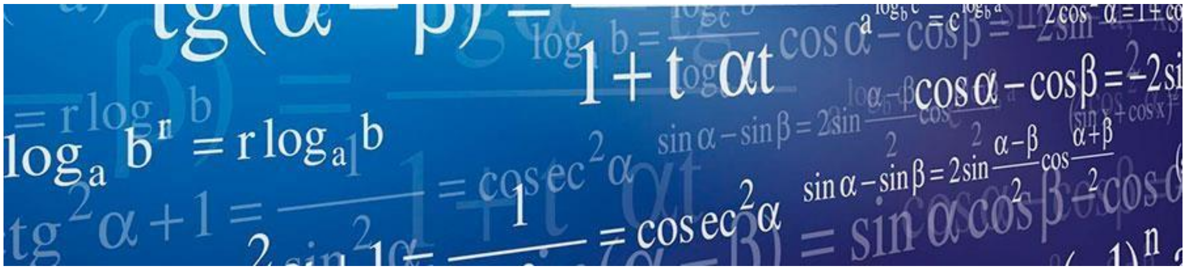


EJERCICIOS DE ORDEN DE COMPLEJIDAD TEMPORAL Y ESPACIAL

Escribe la función de complejidad temporal T(n) y complejidad espacial E(n) para cada uno de los siguientes algoritmos, así como su orden de complejidad (Notación O).

1. Encuentre el orden O de complejidad temporal y espacial del algoritmo de ordenamiento por Burbuja Simple.

```
Procedimiento BurbujaSimple(A,n)
    para i=1 hasta (i<n) hacer
        para j=0 hasta (j<n-1) hacer
            si (A[j]>A[j+1]) hacer
                temp = A[j]
                A[j] = A[j+1]
                A[j+1] = temp
            fin si
        fin para
    fin para
fin Procedimiento
```

Temporal :
$O(n) = n^2$
Espacial :
$E(n) = n+4$

2. Encuentre el orden O de complejidad temporal y espacial del algoritmo de ordenamiento por Inserción.

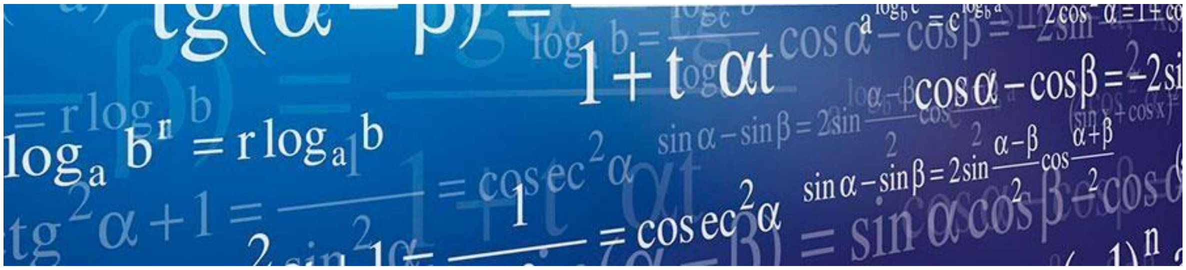
```
Procedimiento Insercion(A,n)
{
    para i=1 hasta i<n hacer
        temp=A[i]
        j=i-1
        mientras((A[j]>temp)&&(j>=0)) hacer
            A[j+1]=A[j]
            j--
        fin mientras
        A[j+1]=temp
    fin parfin Procedimiento
```

Temporal

$O(n) = n^2$

Espacial

$E(n) = n + 4$



3. Encuentre el orden O de complejidad temporal y espacial del algoritmo de ordenamiento por Selección.

```
Procedimiento Seleccion(A,n)
    para k=0 hasta k<n-1 hacer
        p=k;
        para i=k+1 hasta i>n-1 hacer
            si A[i]<A[p] hacer
                p = i
            fin si
            si p!=k hacer
                temp = A[p]
                A[p] = A[k]
                A[k] = temp
            fin si
        fin para
    fin para
fin Procedimiento
```

Temporal:
$O(n) = n^2$
Espacial:
$E(n) = n + 5$

4. Encuentre el orden O de complejidad temporal y espacial del algoritmo de ordenamiento Shell.

```
Procedimiento Shell(A,n)
   k = n / 2;
   mientras k >= 1 hacer
     para i=k hasta i>=n hacer
       v = A[i]
       j = i - k;
       mientras j >= 0 && A[j] > v hacer
         A[j + k] = A[j];
         j  -=  k;
       fin mientras
       A[j + k] = v;
     fin para
    k/=2;
    fin mientras
fin Procedimiento
```

Temporal:
$O(n) = (n\,log\,2\,n)$
Espacial:
$E(n) = n + 5$

5. El máximo común divisor de dos enteros positivos *n* y *m*; denotado por *MCD(n,m)*; es el único entero positivo *k* tal que *k* divide a *m* y *n* y todos los demás enteros que dividen a *m* y *n* son menores que *k*. Encuentre el orden *O* de complejidad temporal y espacial del algoritmo.

```
func MaximoComunDivisor(m, n)
{
    a=max(n,m);
    b=min(n,m);
    residuo=1;
    mientras (residuo > 0)
    {
        residuo=a mod b;
        a=b;
        b=residuo;
    }
    MaximoComunDivisor=a;
    return MaximoComunDivisor;
}
```

Temporal :

$O(n) = n$

Espacial :

$E(n) = n + 4$