Programación 2
Práctico 6 – Análisis de Algoritmos

1 – Realizar un informe sobre **Análisis de Algoritmos,** que contenga los siguientes temas:

- Concepto de Eficiencia de algoritmos en términos de tiempo de ejecución
- Eficiencia: análisis exacto versus análisis estimado
- Análisis estimado: Conceptos fundamentales y Noción de Orden de un algoritmo.
- Análisis de algoritmos sobre estructuras de datos conocidas (arreglos, listas, árboles, grafos). Ver algoritmos de búsqueda.

2 – Para cada uno de los siguientes algoritmos, determine:

- Qué hace
- La operación básica
- El tamaño de la entrada
- El mejor y el peor caso
- Para el mejor caso, calcular la cantidad de veces que se ejecuta la operación básica
- Idem para el peor caso
- El orden

a)
```
void Mostrar (Matriz M)
{
int i,j ;
 for (i=0 ; i<TAM ; i++)
  for (j=0 ; j<TAM ; j++)
   printf ("%d",M[i][j]);
}
```

b)
```
void Borrar (int x, ArrayConTope &A)
{
 int i=0;
 A.tope--;
 while((A.arr[i]!=x)&&(i<A.tope))
  i++;
 A.arr[i] = A.arr[A.tope];
}
```

c)
```
int Obtener (Arreglo a, int pos)
{
int help;
help = a[pos];
return help;
}
```

d)
```
void Desplegar (int x , int n)
{   int i,j;
    for ( i=0 ; i<n+1 ; i++)
    {
j=0 ;
          for (j=0 ; j<n+2 ; j++)
           printf ("%d\n",x);
      }
   }
```

e)
```
int Contar (Arreglo a, int e)
{
int i,cont=0;
 for(i=0; i<TAM; i++)
  if (a[i]==e)
   cont++;
 return cont;
}
```

f)
```
int Largo (Lista L)
{
  int cont=0;
  while (L != NULL)
  {
        cont++;
   L = L -> sig;
  }
  return cont;
 }
```

g)
```
boolean Pertenece (ABB a, int e)
{
 boolean esta = false;

 while (a != NULL) && (!esta)
 {
  if (e == a -> info)
   esta = true;
  else
   if (e < a->info)
    a = a -> hizq;
   else
    a = a -> hder;
 }

 return esta;
}
```

h)
```
void DFS (Grafo G, int verticeActual, boolean visitado[N])
{
visitado[verticeActual] = true;
  printf ("%d", verticeActual);
  for (int j=0; j<N; j++)
{
}

 if (G[verticeActual][j] == 1)
 {
if (!visitado[j])
   DFS(G, j, visitado);
 }
}
```