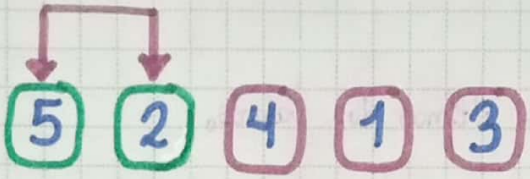
FECHA

TEMA

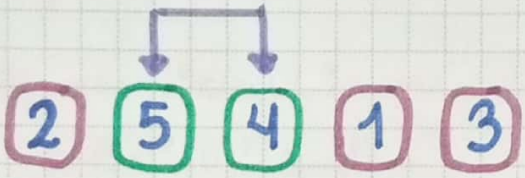
Algoritmo de ordenación de burbuja

Comenzamos con una lista de elementos no ordenados.

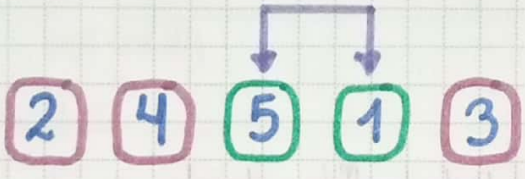
Tomamos los primeros dos números y si no están ordenados se intercambian los lugares.



Se repite el proceso con los siguientes dos números.



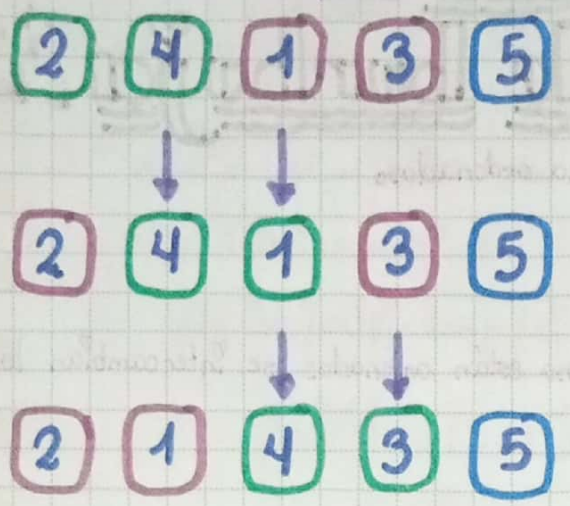
El proceso continúa hasta llegar al final.



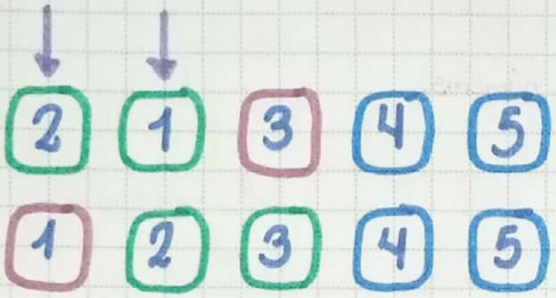
FECHA

TEMA

El último número ya queda ordenado por lo que en la siguiente iteración ya no se evalúa acortando el proceso.



En la tercera iteración no se evalúan los últimos dos valores.



La cuarta iteración se finaliza sin que se haya realizado un intercambio por lo que el algoritmo termina.

Al finalizar el algoritmo tenemos como resultado la lista ordenada.

Programa

```
#include <iostream>

using namespace std;
void ordenacion(int v[], int tam)
{
    for(int i=0; i<=tam; i++)
    {
        for(int j=0; j<tam-1; j++)
        {
            int aux;
            if(v[j] > v[j+1])
            {
                aux = v[j+1];
                v[j+1] = v[j];
                v[j] = aux;
            }
        }
    }
}
void mostrar(int v[], int tam)
{
    for(int i=0; i<tam; i++)
    {
        cout << v[i] << " ";
    }
    cout << endl;
}
int main()
{
    int v[10] = {3, 4, 22, 76, 1, 90, 0, 6, 13, 102};
    ordenacion(v, 10);
    mostrar(v, 10);
    return 0;
}
```

Algoritmo de búsqueda

Búsqueda secuencial.

Este tipo de búsqueda se aplica aunque no esté ordenado el arreglo.

Dependiendo del planteamiento del problema y si haya datos repetidos, podrá encontrar el primer dato y el registro completo (struct) y devolver la posición, un valor booleano, indicar que sí se encontró, las posiciones donde se encontró un arreglo con todos los registros encontrados, etc.

Búsqueda secuencial.

```
int busquedaSecuencial(Alumno alumnos[], int tam,
char nombre[]){
int i, band=0;
for (i=0; i<tam && band==0; i++){
    if (strcmp(alumnos[i].nombre, nombre)==0)
        band=1;
}
return band;
}
```

Búsqueda secuencial & ejercicio.

Dado un arreglo alumnos[] = {{4, "Juan", 8.2}, {2, "Ana", 7.2}, {3, "Juan", 6.5}, {1, "Hector", 7.2}, {5, "Ana", 8.2}};

Escriba las siguientes funciones de búsqueda secuencial con los criterios que se les indican:

1o Por promedio, devolviendo los registros correspondientes.
2o Por matrícula, devolviendo el registro correspondiente.
3o Por nombre, devolviendo el número de alumnos encontrados.

FECHA

TEMA

TOP

Búsqueda binaria

Este tipo de búsqueda se aplica solo si el arreglo está ordenado.

Por lo tanto, solo debe encontrar una coincidencia en caso que se encuentre el dato.

Podría devolver un valor booleano, la posición de dato o un registro.

Búsqueda binaria & algoritmo.

```
int busquedaBinaria (Alumno lista[], int tam, int matricula){
int i=0, j=tam-1
do{
      medio =((i+j)/2);
      if (matricula > lista[medio].matricula)
          i= medio+1;
      else
          j= medio-1;
}while (alumnos[medio].matricula != matricula && i<=j);
if (matricula != alumnos[medio].matricula)
     medio=-1;
return medio;
}
```

Búsqueda secuencial & ejercicio.

Dado un arreglo alumnos[]={{4,"Juan",8.2},{2,"Ana",7.2},{3,"Juan",6.5},{1,"Hector",7.2},{5,"Ana",8.2}};

Escriba una función que reciba una matrícula y devuelva el registro completo.

FECHA

TEMA

Algoritmo y estructura de datos

Definición.

- Un archivo es una colección de datos que con un título o nombre se almacena o graban en dispositivos tales como discos, diskettes, cintas magnéticas, usb, ssd, dvd, etc.

Operaciones básicas con archivos.

- Apertura del archivo empleando un nombre para identificarlo.
- Escritura de datos en el archivo.
- Lectura de datos del archivo.
- Cierre del archivo.
- Los datos de un archivo son simplemente bytes o caracteres almacenados uno a continuación de otro que son enumerados con 0, 1, 2, sucesivamente.

Funciones de alto nivel.

- Se encuentran en la librería <stdio.h>
- Para usar estas funciones se debe declarar una variable apuntador del tipo predefinido FILE en la forma:
- FILE *arch;
- En donde arch es su nombre o identificador.

Modos de abrir un archivo.

- "w" para escribir.
- "r" para leer.
- "a" para añadir.
- fopen() se utiliza para abrir un archivo y toma valores apuntadores de tipo FILE en la forma:

```
arch=fopen("nombre de archivo",modo);
```

- fclose() se utiliza para cerrar un archivo despues de haber realizado alguna operación.
  Finaliza el vinculo con la variable arch.

Escritura de archivos

- Para escribir en un archivo (abierto en los modos "w" o "a") se pueden utilizar las funciones:
  - fprintf (arch, "cadena de control", lista de datos);
  - fputc (c, arch); escribe un carácter c en arch.
  - fputs (s, arch); escribe una cadena en arch hasta encontrar su final ('\0');
  - fwrite (s, m, n, arch); escribe en arch, n datos de tamaño o longitud m ubicados en la dirección s.

Lectura de archivos.

- fscan (arch, "cadena de formato", lista de direcciones); igual que scanf pero su lectura se hace en arch.
- fgetc (arch); devuelve el siguiente carácter leído en arch.
- fgets (s, n, arch) lee una línea de caracteres, hasta un maximo de n-1 caracteres, hasta un maximo de n-1 caracteres o hasta encontrar un cambio de línea del archivo arch y los almacena en la dirección s. Añade al final el caracter nulo ('\0').

- fread (s, m, n, arch); lee en arch n datos de tamaño o longitud m (igual a m*n bytes) y los almacena a partir de la dirección s.
- feof (arch); que sirve para comprobar si se esta al final del archivo (valor distinto de cero) o no (valor cero).