Informe de la parte individual sobre el algoritmo bubble sort implementado en ensamblador.

Mi implementación usa de 2 bucles, uno interno y otro externo que van desde la primera posición del array hasta el final.

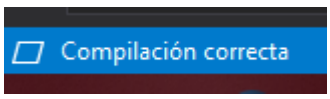
Ambos usan de 2 contadores, que se inicializan con la longitud del array - 2, y van decrementando hasta que sean 0.

Van comparando la posición actual con la siguiente, en caso de que sea mayor la actual que la siguiente, se les da la vuelta.

Y para pasar a la siguiente iteración, como el array es de enteros (Int), sumamos 4 al registro de la posición actual del vector.

Cuando el bucle interno acaba, decrementa en 1 el externo, y el contador interno se resetea, así hasta que el contador externo sea 0.

```
void bubbleSort(int* arr, int size)
{
    int a = size - 2;           // tamaño - 2 porque estámos comporbando cada 2 grupos a la vez, por lo tanto, el ultimo par no se comprueba
                                //     ya que si se hace, se saldria fuera del rango del vector, y comprobaria la ult. pos. con algo
                                //     que no hace parte del vector.

    __asm {
        mov esi, arr        // Asingamos la dir del inicio del array a esi.
        mov edi, a;         // Asignamos el tamaño del array a edi.
        mov eax, edi;       // eax = contador externo.

    externo:                // Tag del bucle externo.
        mov ebx, edi;       // ebx = contador interno.

    interno:                // Tag del bucle interno.
        mov ecx, [esi];     // ecx = valor de la pos. actual del vector.
        mov edx, [esi + 4]; // edx = Valor posición siguiente.
        cmp ecx, edx;       // Comparamos el valor actual con el siguiente, el resultado e
        jl guardar;         // Si el valor actual ECX es menor que la siguiente [ESI + 4], entonces nos saltamos el paso de intercambiarlos.

        xchg ecx, edx;      // Intercambiamos.

    guardar:                // Tag  de las operaciones comunes por iteracion.
        mov [esi], ecx;     // Guardamos el valor actual.
        mov [esi + 4], edx; // Guardamos el valor siguiente.

        add esi, 4;         // Incrementamos el registro en 4 para pasar a la siguiente posición del vector.

        dec ebx;            // Decretemntamos 1 en el contador interno.
        jnz interno;        // Si no es 0, repetimos el bucle interno.

        mov esi, arr        // Re-asingamos la dir del inicio del array a esi
        dec eax;            // Decrementamos 1 en el contador externo.
        jnz externo;        // Si no es 0, repetimos el bucle externo.

    }
}
```




```
Consola de depuración de Microsoft Visual Studio
Valor del array desordenado:
22722, 18670, 14726, 21273, 30779, 22388, 8541, 21138, 278, 8786, 7864, 5605, 1333, 10992, 18477, 18653, 29317, 5017, 12967, 31053

Valor del array ordenado:
278, 1333, 5017, 5605, 7864, 8541, 8786, 10992, 12967, 14726, 18477, 18653, 18670, 21138, 21273, 22388, 22722, 29317, 30779, 31053

C:\Users\Anubis\Desktop\UA\UAADCProjects\Individual\Parte1\EjerciciosEnsamblador\Debug\Ejemplo1.exe (proceso 13152) se cerró con el cód
igo 0.
Para cerrar automáticamente la consola cuando se detiene la depuración, habilite Herramientas ->Opciones ->Depuración ->Cerrar la cons
ola automáticamente al detenerse la depuración.
Presione cualquier tecla para cerrar esta ventana. . .
```