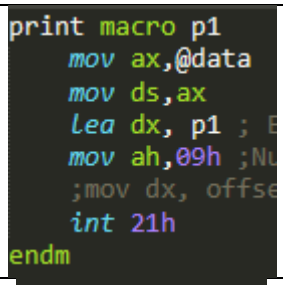
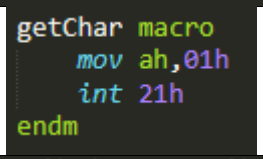
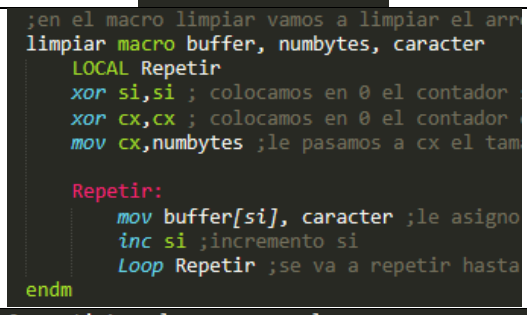
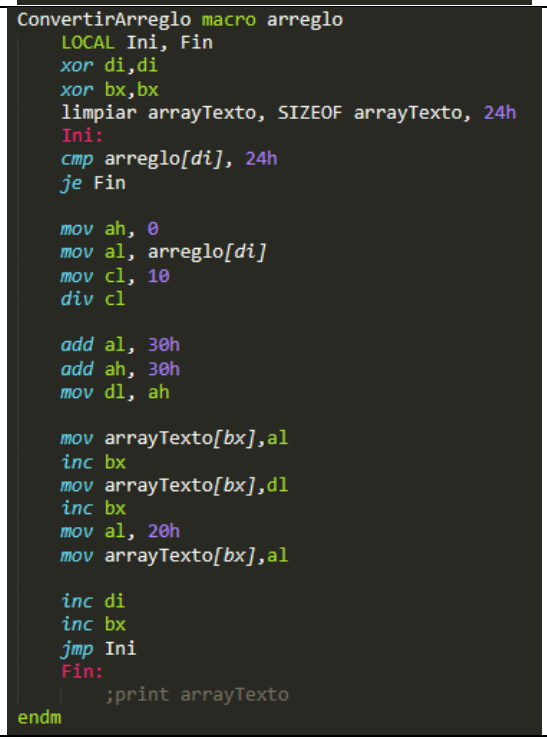
UNIVERSIDAD SAN CARLOS DE GUATEMALA
ARQUITECTURA DE COMPUTADORAS Y ENSAMBLADORES 1
SECCIÓN B
PRIMER SEMESTRE 2021

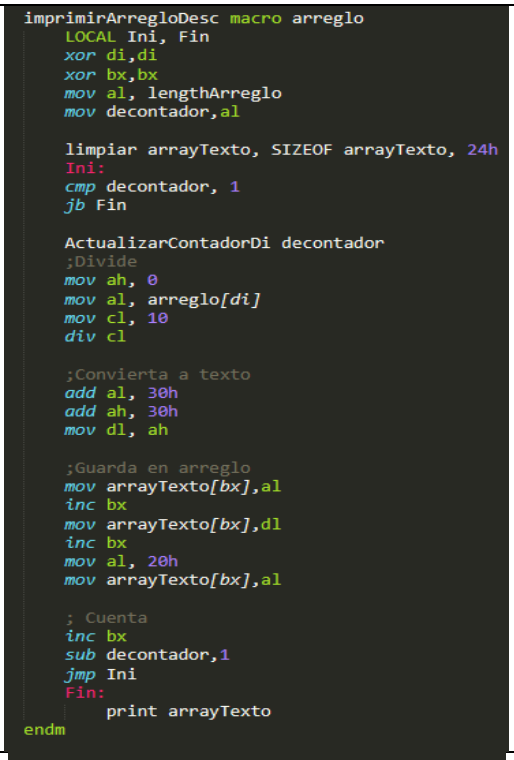
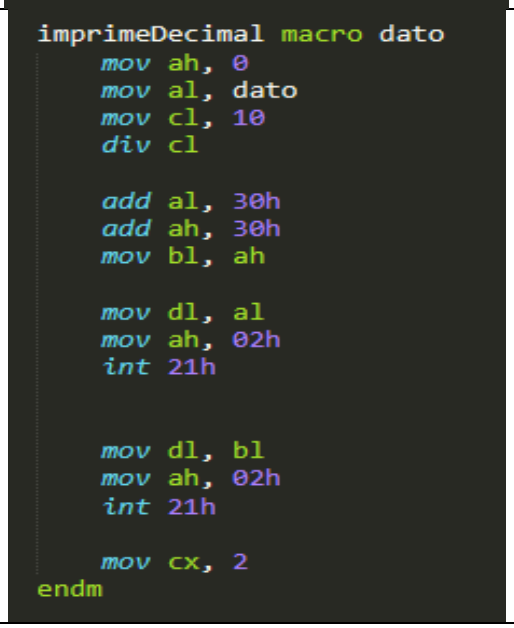
PRÁCTICA 4

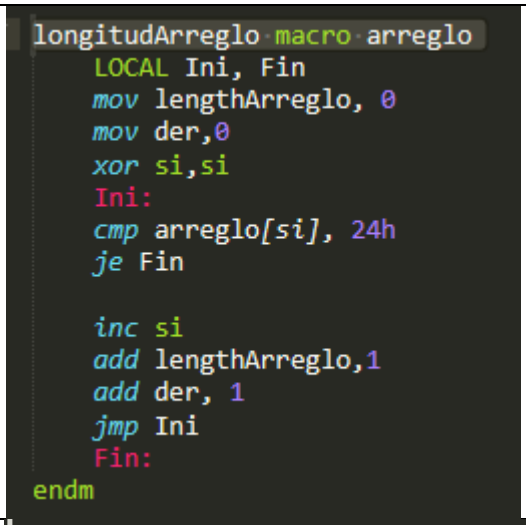
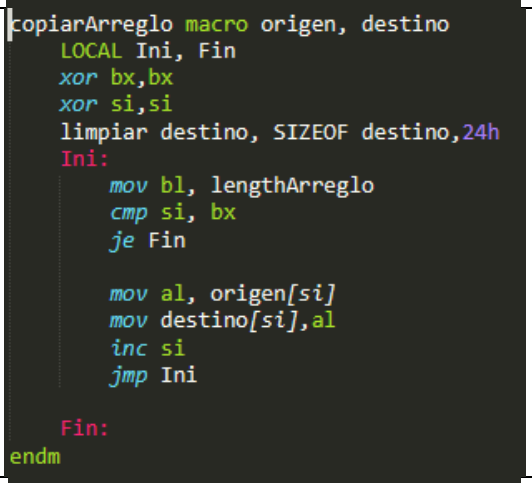
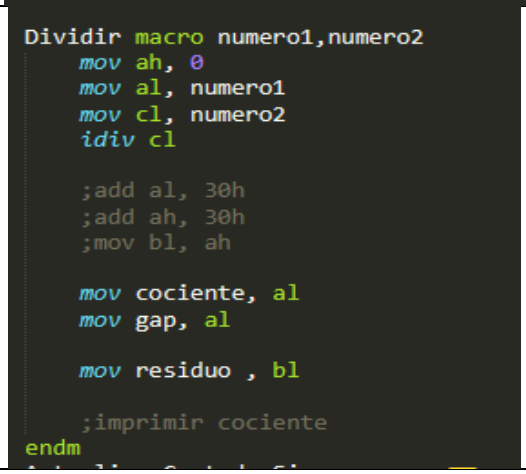
201801263 Audrie Annelisse del Cid Ochoa

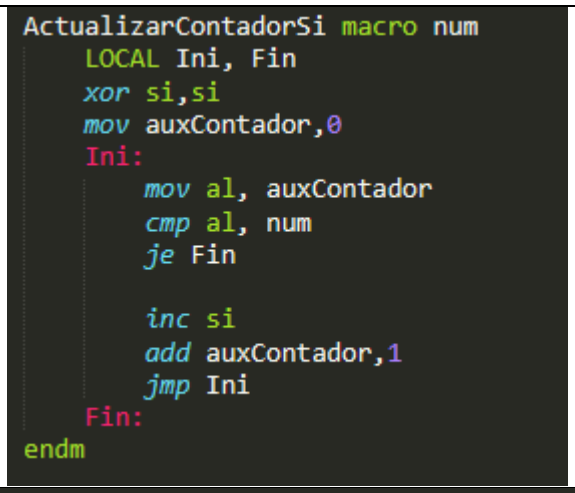
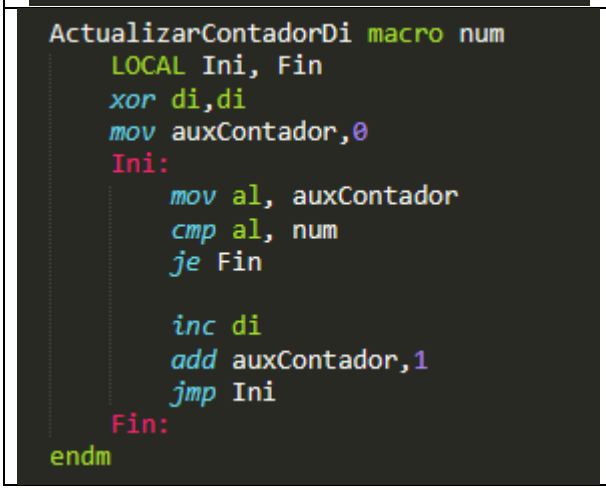
Manual Técnico

Macros utilizados constantemente a lo largo del proyecto:

Macro	Función
	Permite imprimir cadenas, utilizando la interrupción 21h
	Permite obtener un carácter. Se utilizó para obtener los datos al escoger una opción. Su valor queda guardado en al.
	Limpiar, fue utilizado para limpiar arreglos. Permitía llenarlos con el carácter que se escogiera, en la mayoría de los casos fue el signo de dollar.
	Convertir Arreglo, fue utilizado para convertir un arreglo de enteros a uno de texto.

		Imprimir arreglo, fue utilizado para imprimir un arreglo de enteros.
		ImprimeDecimal, convierte los datos ascii en enteros y los imprime en pantalla.

			longitudArreglo, permite obtener el tamaño de un arreglo y lo guarda en la variable lenghtArreglo.
			CopiarArreglo, permite mover todos los datos de un arreglo a otro.
			Dividir, realiza una división de enteros

	ActualizarcontadorSi, permite obtener el registro índice si, en el número deseado mediante un parámetro entero.
	Realiza la misma función que el anterior, pero para el registro índice di.

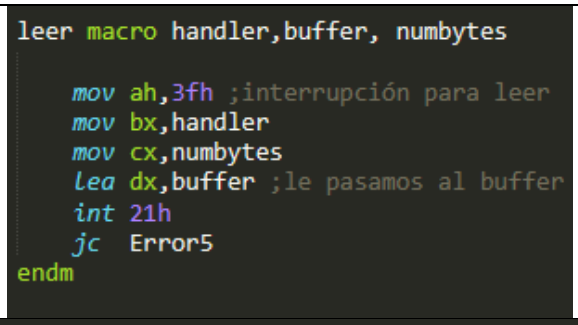
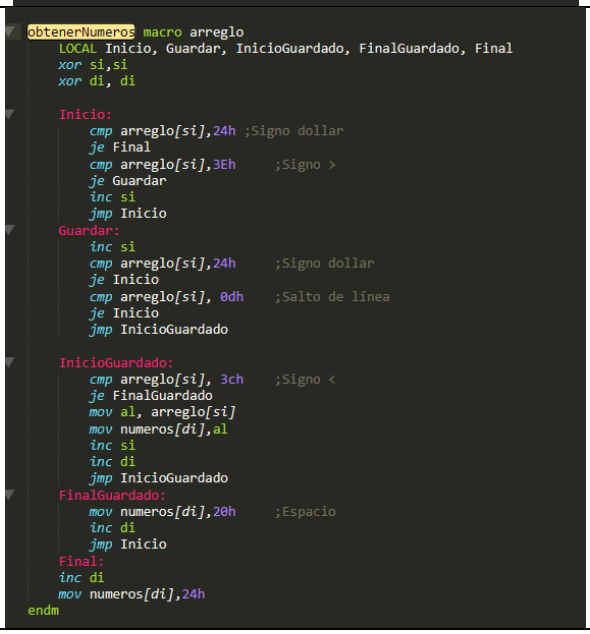
1. Cargar Archivo

```
CargarArchivo:
        SaltoLinea
        print menu01
        call AbrirArchivo

        jmp MenuPrincipal
AbrirArchivo:
        print carga1
        SaltoLinea
        limpiar numeros, SIZEOF numeros,24h
        limpiar bufferentrada, SIZEOF bufferentrada,24h
        obtenerRuta bufferentrada
        abrir bufferentrada, handlerentrada
        limpiar bufferInformacion, SIZEOF bufferInformacion,24h
        leer handlerentrada, bufferInformacion, SIZEOF bufferInformacion
        call cerrarArchivo
        obtenerNumeros bufferInformacion
        convertirNumero numeros  ;Resultados en el arreglo
        longitudArreglo numeroReal
        print finCarga
```

Se utilizaron los siguientes Macros:

Macro	Función

`obtenerRuta macro buffer`
`    LOCAL ObtenerChar, endTexto`
`    xor si,si ; xor si,si = mov si,0`
 	
`    ObtenerChar:`
`        getChar`
`        cmp al,0dh ; ascii de salto de linea en hex`
`        je endTexto`
`        mov buffer[si],al ;mov destino, fuente`
`        inc si ; si = si + 1`
`        jmp ObtenerChar`
 	
`    endTexto:`
`        mov al,00h ; asci del caracter nulo`
`        mov buffer[si], al`
`endm`
	Obtener Ruta, este macro funciona similar a obtener texto, dado que permite ingresar una serie de caracteres y termina en cuando encuentre un salto de línea. Fue visto en clase.

			Leer, permite leer el arreglo de entrada los cuales contienen los datos que estaban dentro del documento que se ha abierto.
			Este macro fue utilizado para ir leyendo el arreglo de entrada que contiene los datos del documento y conforme va avanzando, los analiza y guarda en un arreglo de números ascii.

```
convertirNumero macro arreglo
    LOCAL Inicio, Guardar, Final
    mov contadorArreglo,0
    xor si,si
    xor di,di
    limpiar valorNumero, SIZEOF valorNumero, 24h

    Inicio:
        ;imprimir arreglo[si]
        cmp arreglo[si], 24h    ;Signo dollar
        je Final

        cmp arreglo[si],20h     ;Espacio
        je Guardar

        mov al, arreglo[si]
        mov valorNumero[di],al
        inc di
        inc si
        jmp Inicio
    Guardar:
        mov di, contadorArreglo
        mov auxsi, si
        mov auxdi, di
        obtenerSizeBuffer valorNumero
        filtarNumero valorNumero
        ;StringToInt valorNumero

        mov al, numero2
        ;imprimeDecimal numero2

        mov numeroReal[di],al
        limpiar valorNumero, SIZEOF valorNumero, 24h
        mov si, auxsi
        mov di, auxdi
        inc si
        xor di,di
        inc contadorArreglo
        jmp Inicio
    Final:
        ;imprimeDecimal numeroReal[0]
```

Este macro permite convertir los números del arreglo utilizado para guardar los números obtenidos del análisis del archivo cargado y los convierte en números decimales.

2. Ordenar
    a. Método Burbuja

```
Burbuja:
    print msjVelocidad
    getChar
    sub al, 48
    mov velocidadBubble, al
    SaltoLinea
    add velocidadBubble, 30h
    ;print velocidadBubble
    print menu5
    print msjAscendente
    print msjDescendente
    SaltoLinea
    getChar
    cmp al, 31h
    je BurbujaAscendente

    cmp al, 32h
    je BurbujaDescendente
```

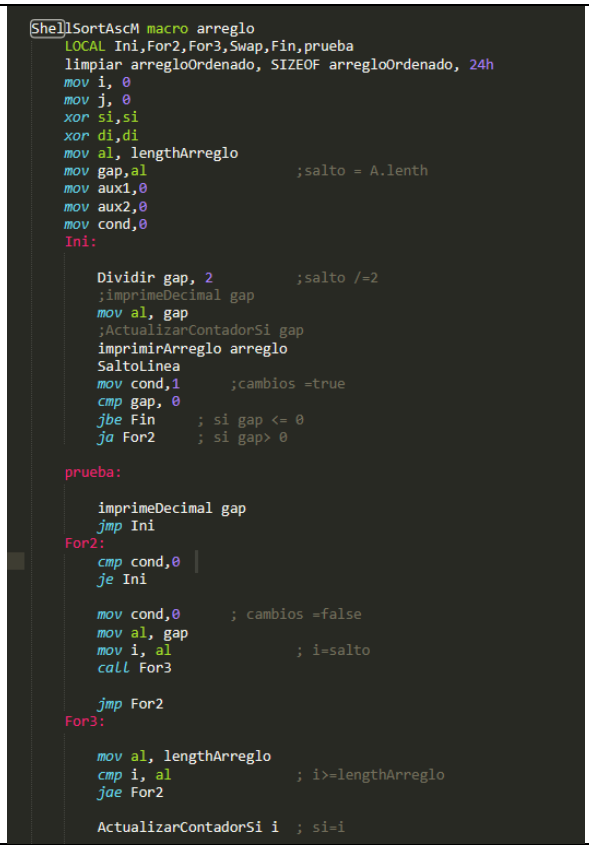
Macro	Función

```
BurbujaAsc macro arreglo
    Local Ini,For2, Fin, Swap
    limpiar arregloOrdenado, SIZEOF arregloOrdenado, 24h
    mov aux, 0
    mov i,1
    mov j, 0
    xor si,si
    Ini:
        add i, 1
        mov j, 0
        xor si,si

        imprimirArreglo arreglo
        SaltoLinea
        ;Graficar arreglo
        mov al, lengthArreglo
        cmp i, al
        jae Fin      ; si i >= lengthArreglo
        jb For2      ; si i< lengthArreglo


    For2:
        mov al, lengthArreglo
        sub al, 1    ; lengthArreglo-1

        cmp j, al
        jae Ini      ; j>= lengthArreglo-1

        mov bl, arreglo[si]


        cmp bl,arreglo[si+1]
        ja Swap      ;arreglo[si] > arreglo[si+1]



        add j, 1
        inc si
        jmp For2
    Swap:


        mov al, arreglo[si]      ; aux = arreglo[si]
        mov aux, al

        mov al, arreglo[si+1]       ; al = arreglo[si+1]


        mov arreglo[si],al       ; arreglo[si] = arreglo[


        mov al, aux              ; al = aux
        mov arreglo[si+1], al    ; arreglo[si +1 ]= aux
```

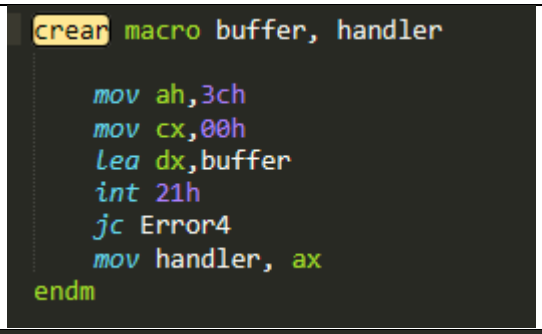
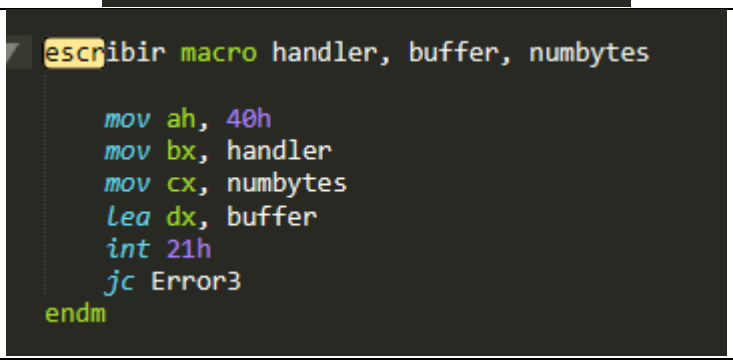
Este macro permite realizar el ordenamiento burbuja de forma ascendente, se planteo igual que una parte de código java del mismo método. También se realizó un procedimiento para ordenar los datos de forma descendente, en ese caso sólo cambió la condición de salto en la parte del swap. De ja pasó a jb para indicar menor.

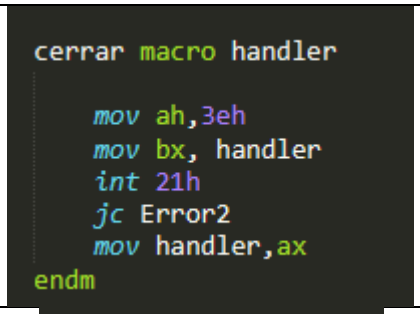
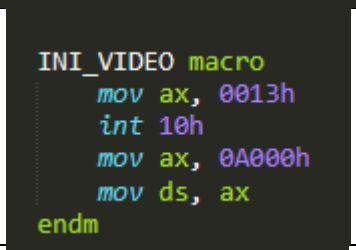
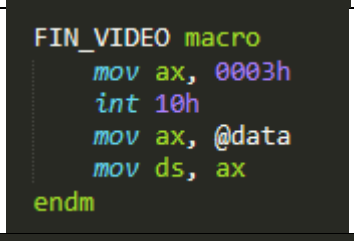
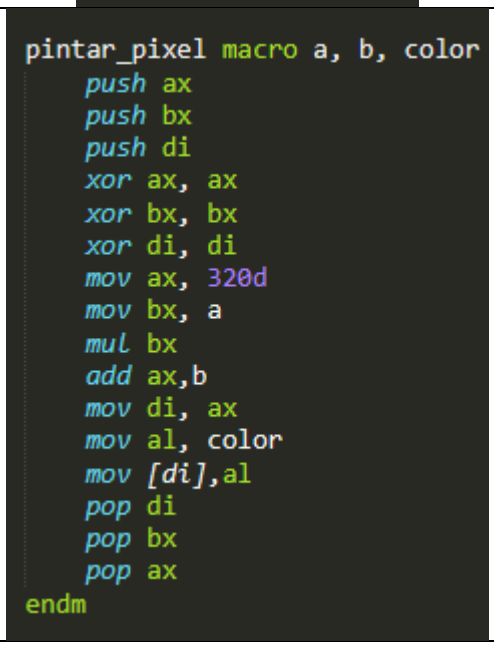
```
QuickSortAscP proc

      mov cx,der
      cmp izq,cx
      jge fin
      partitionAsc arrayQuick, izq, der
      mov bx, der
      push bx
      mov ax, resultadoPartition
      push ax
      dec ax
      mov der,ax

     imprimirArreglo arrayQuick
     SaltoLinea
            ;tienen que graficar aqui

      call QuickSortAscP
      pop ax
      pop bx
      mov der,bx
      inc ax
      mov izq,ax
      call QuickSortAscP
      fin:
          ret
QuickSortAscP endp
```

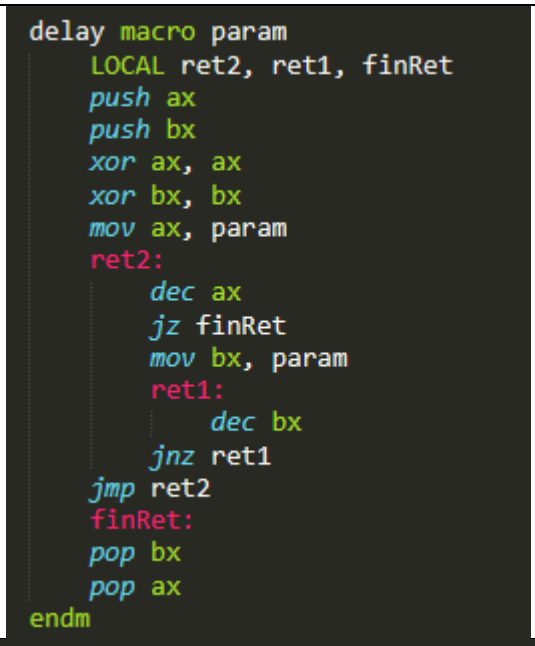
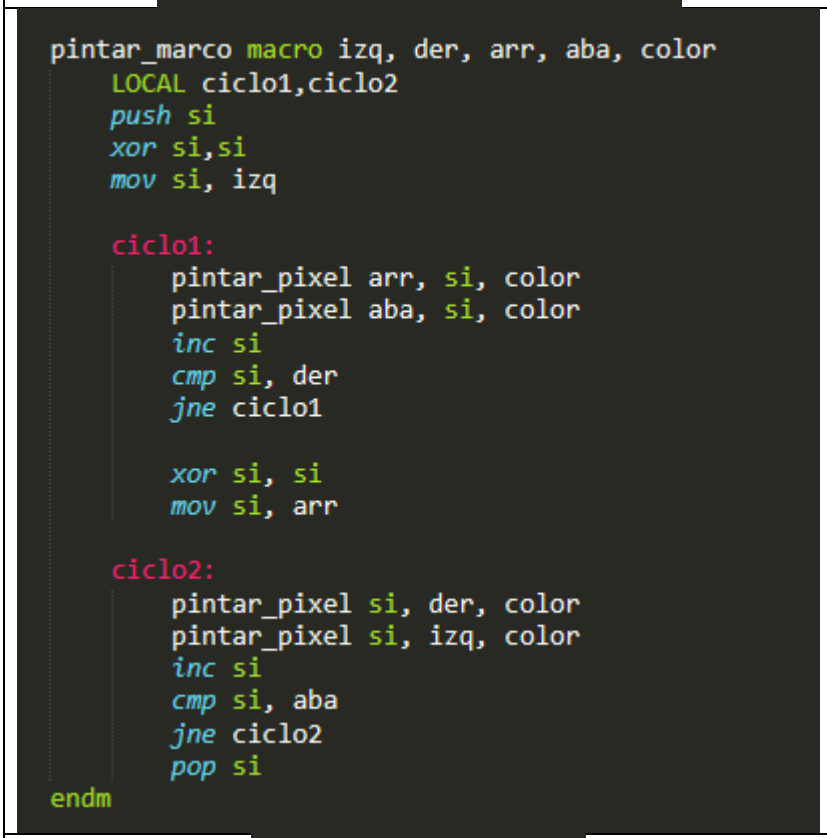
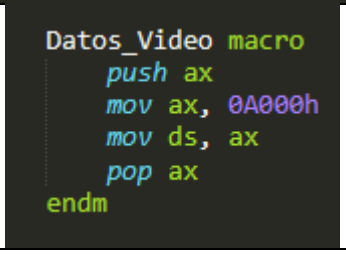
Para el procedimiento Quicksort visto en clase se utilizó procedimiento en vez de un macro, dado que era recursivo.

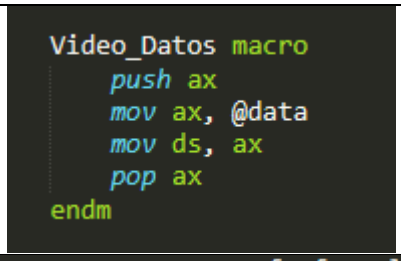
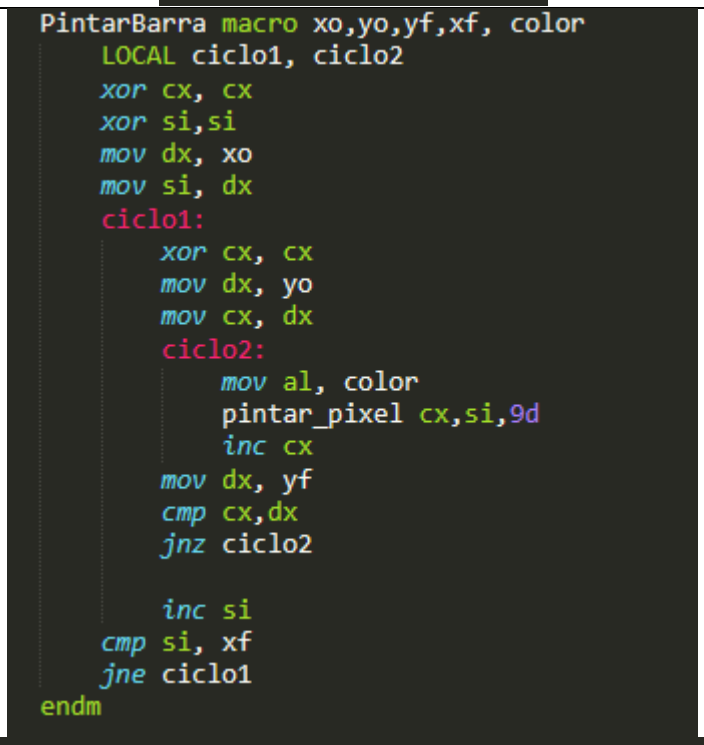
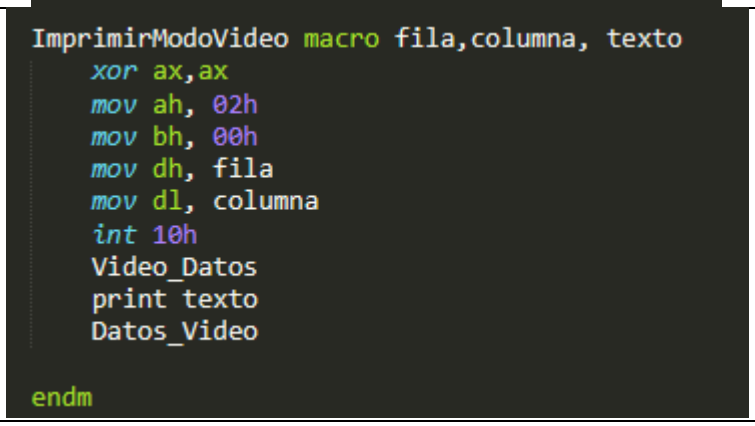
	Para el macro shellSort se guió por un ejemplo visto en internet en código java.

3. Generar Reporte

Macro	Función
	Permite crear un archivo.
	Permite escribir texto dentro del archivo creado y abierto.

		Permite cerrar el archivo abierto.
		Permite iniciar el modo video. Cambia de modo texto a modo video.
		Permite finalizar el mod video y cambia a modo texto
		Permite pintar un pixel en la pantalla, teniendo un lienzo de 320*200 pixeles. Se realiza mediante coordenadas y eligiendo un color.

	Delay, permite generar un retardo mediante la simulación de un bucle.
	Pintar_macro, permite pintar marcos mediante medidas de inicio en el eje x, fin en el eje x, inicio en el eje y , fin en el eje y. Así como el color.
	Datos_video, permite pasar de modo video a modo Texto, sin necesitad de sacarte de la aplicación.

		Video_Datos, permite pasar de modo video a modo texto sin sacarte de la aplicación.
		Pintar Barra, permite generar barras mediante coordenadas en diferentes posiciones del lienzo.
		Imprimir Modo Video, permite

4. Salir