Elian Saúl Estrada Urbina
201806838

# MANUAL TÉCNICO

1. **PROC'S**
   - **a.**
     - i. **Nombre:** Main
     - ii. **Proposito:** Procedimiento principal del segmento de datos
     - iii. **Codigo:**

```
main proc

    mov dx, @DATA
    mov ds, dx
    mov es, dx

    MENU:
        clear

        mov flagMenu, 30h

        print headerMsg
        print menuMsg

        inputOptions flagTrue

        cmp al, 31h
        je FILE_UPLOAD
        cmp al, 32h
        je VELOCITYOP
        cmp al, 33h
        je GENERA_REPORT
        cmp al, 34h
        je EXIT
        jne MENU_ERROR

    MENU_ERROR:
        clear

        print menuErrorMsg
        print keyPress
        inputOptions flagTrue

        chooseMenu flagMenu


    FILE_UPLOAD:
        print fileUploadMsg
        print pathFileMsg

        getInput bufferFileName

        openFile bufferFileName
        readFile
        closeFile


        getListNumber bufferFile, bufferList, bufferNumber

        copy bufferList, bufferListB

        jmp MENU
```

Elian Saúl Estrada Urbina
201806838

```
VELOCITYOP:

    print velocityMenu

    inputOptions flagTrue

    xor dx, dx
    mov dl, al
    mov lblVelocityV, al

    sub dl, 48d

    xor ax, ax
    mov ax, dx
    mov dx, 300d

    ;mov bx, 200d
    mul dx
    mov bx, 3000d
    sub bx, ax

    mov velocity, bx
    mov ax, bx


ORMENU:

    clear

    mov flagMenu, 32h

    print headerMsg
    print orMenuMsg

    inputOptions flagTrue

    cmp al, 31h
    jb MENU_ERROR
    cmp al, 32h
    ja MENU_ERROR

    mov flagTypeSort, al

SORT:

    clear

    mov flagMenu, 31h

    print headerMsg
    print sortMsg

    inputOptions flagTrue

    cmp al, 31h
    je BUBBLESORT
    cmp al, 32h
    je QUICKSORT
    cmp al, 33h
    je SHELLSORT
    jne MENU_ERROR
```

Elian Saúl Estrada Urbina
201806838

```
BUBBLESORT:

    call mode_video

    paintBorder 20d, 195d, 2d, 317d, 15d
    ;paintBar 25d, 180d, 10d, 20d, 15d

    call seg_text

        posCursor 1d, 1d
        print lblBubbleSort

        posCursor 1d, 13d
        print lblTime

        posCursor 1d, 25d
        print lblVelocity

        posCursor 1d, 35d
        print lblVelocityV

        call calcWidth

        ;mov currentValue, 25d


        ;call calcHeight

        call drawGraph
        delay velocity
        bubbleSortL bufferList

    call seg_video


    delay 3000

    call mode_text

    jmp MENU
```

Elian Saúl Estrada Urbina
201806838

```
GENERA_REPORT:

    createFile reportFile
    writeFile reportHeader
    writeFile reportHeader2
    writeFile reportHeader3
    call getDate
    mov al, day
    getBuffer bufferDate
    writeFile lblDay0
    writeFile bufferDate
    writeFile lblDayC
    mov al, month
    getBuffer bufferDate
    writeFile lblMonth0
    writeFile bufferDate
    writeFile lblMonthC
    mov ax, year
    getBuffer bufferDate
    writeFile lblYear0
    writeFile bufferDate
    writeFile lblYearC
    writeFile lblHour0
    writeFile lblEnter
    clearBuffer bufferDate
    call getTime
    mov al, hour
    getBuffer bufferDate
    writeFile lblHour0
    writeFile bufferDate
    writeFile lblHourC
    mov al, minute
    getBuffer bufferDate
    writeFile lblMinute0
    writeFile bufferDate
    writeFile lblMinuteC
    mov al, second
    getBuffer bufferDate
    writeFile lblSecond0
    writeFile bufferDate
    writeFile lblSecondC
    writeFile lblStuden
    writeFile lblResult
    call getTypeSort
    writeFile lblTypeC
    writeFile lblInputList0
    showArray bufferListB
    writeFile lblInputListC
    writeFile lblSortList0
    showArray bufferList
    writeFile lblSortListC
    writeFile lblBubbleSort0
    writeFile lblVelocity0
    writeFile lblVelocityV
    writeFile lblVelocityC
    writeFile lblBubbleSortC
    writeFile lblResultC
    writeFile lblHeaderC

    closeFile

    print keyPress
    inputOptions flagFalse
```

Elian Saúl Estrada Urbina
201806838

```
EXIT:
    print goodbyeMsg

    mov ah, 4ch
    int 21h
```

**b.**

i. **Nombre:** mode_video
ii. **Propósito:** Cambiar el modo de texto por el de video
iii. **Código:**

```
mode_video proc

    mov ax, 13h
    int 10h

    mov dx, 0A000h
    mov ds, dx
    ret

mode_video endp
```

**c.**

i. **Nombre:** mode_text
ii. **Propósito:** Cambiar el modo de video por el de texto
iii. **Código:**

```
mode_text proc

    mov ax, 03h
    int 10h

    mov dx, @DATA
    mov ds, dx
    mov es, dx

    ret

mode_text endp
```

Elian Saúl Estrada Urbina
201806838

**d.**

i. **Nombre:** seg_video
ii. **Propósito:** Cambiar la memoria de texto por la de video
iii. **Código:**

```
seg_video proc

    push dx

    mov dx, 0A000h
    mov ds, dx

    pop dx

    ret

seg_video endp
```

**e.**

i. **Nombre:** seg_text
ii. **Propósito:** Cambiar la memoria de video por la de texto
iii. **Código:**

```
seg_text PROC

    push dx

    mov dx, @DATA
    mov ds, dx

    pop dx

    ret

seg_text ENDP
```

Elian Saúl Estrada Urbina
201806838

**f.**

i. **Nombre:** clearGraph
ii. **Propósito:** Limpiar el área del grafico para pintar de nuevo
iii. **Código:**

```
clearGraph PROC

    push ax
    push cx
    push dx

    mov ax, 0600h
    mov ch, 3d
    mov cl, 1d
    mov dh, 23d
    mov dl, 38d
    int 10h


    pop dx
    pop cx
    pop ax

    ret

clearGraph ENDP
```

**g.**

i. **Nombre:** calcWidth
ii. **Propósito:** Calcular el ancho de las barras según la cantidad de estas en la pantalla
iii. **Código:**

```
calcWidth PROC

    push ax
    push bx
    push cx
    push dx

    ; xor ax, ax
    ; xor bx, bx
    ; xor cx, cx

    mov cx, countList
    sub cx, 0001h

    mov ax, cx
    mov dx, 5d

    mul dx
    mov cx, ax

    xor ax, ax

    mov ax, 310d
    sub ax, cx

    mov bx, countList

    div bx

    mov maxWidth, ax

    pop dx
    pop cx
    pop bx
    pop ax

    ret

calcWidth ENDP
```

Elian Saúl Estrada Urbina
201806838

**h.**

i. **Nombre:** calcHeight

ii. **Propósito:** Calcular la altura de las barras dependiendo del número mayor leido.

iii. **Código:**

```
calcHeight PROC

    push ax
    push cx
    push bx

    xor ax, ax
    xor bx, bx
    xor cx, cx

    mov ax, 140d
    mov bx, currentValue

    mul bx

    mov bx, maxNumber
    div bx

    xor bx, bx

    mov bx, 165d

    sub bx, ax

    mov maxHeight, bx

    pop bx
    pop cx
    pop ax

    ret

calcHeight ENDP
```

Elian Saúl Estrada Urbina
201806838

i.

i. **Nombre:** drawGraph
ii. **Propósito:** Dibujar todas las barras que representan el arreglo, recorre el arreglo y según el rango del número asigna un color para posteriormente imprimirlo.
iii. **Código:**

```
drawGraph PROC

    push si
    push dx
    push ax
    push bx
    push cx

    xor si, si
    xor dx, dx
    xor cx, cx

    mov cx, 8d


    LOOPPRINT:

        cmp si, countList
        je OUT_DRAW

        xor dx, dx

        mov dl, bufferList[si]
        mov currentValue, dx

        call calcHeight

        xor bx, bx
        xor ax, ax
        mov bx, maxHeight
        mov ax, maxWidth
        add ax, cx

        cmp currentValue, 20d
        jbe RED

        cmp currentValue, 40d
        jbe BLUE

        cmp currentValue, 60d
        jbe YELLOW

        cmp currentValue, 80d
        jbe GREEN

        cmp currentValue, 99d
        jbe WHITE

        RED:
            call seg_video
            paintBar bx, 165d, cx, ax, 4d
            jmp INCPOSLEFT

        BLUE:
            call seg_video
            paintBar bx, 165d, cx, ax, 1d
            jmp INCPOSLEFT

        YELLOW:
            call seg_video
            paintBar bx, 165d, cx, ax, 14d
            jmp INCPOSLEFT

        GREEN:
            call seg_video
            paintBar bx, 165d, cx, ax, 2d
            jmp INCPOSLEFT
```

```
        WHITE:
            call seg_video
            paintBar bx, 165d, cx, ax, 15d
            jmp INCPOSLEFT

        INCPOSLEFT:
            call seg_text
            mov cx, ax
            add cx, 5d

            call showNumber

            inc si

            jmp LOOPPRINT

    OUT_DRAW:

        pop cx
        pop bx
        pop ax
        pop dx
        pop si

    ret

drawGraph ENDP
```

**j.**

i. **Nombre:** showNumber

ii. **Propósito:** Muestra los números del arreglo debajo de la barra que le corresponde.

iii. **Código:**

```
showNumber PROC

    push ax
    push bx
    push dx

    ;xor dx, dx
    xor bx, bx

    mov al, cl
    mov bx, 8d

    div bx
    sub ax, 2d
    mov dx, ax

    ;call seg_video
    posCursor 22d, dl
    ;call seg_text

    ;getBuffer bufferNumber
    ;print bufferNumber

    mov al, bufferList[si]

    getBuffer bufferNumber
    print bufferNumber

    ;inputOptions flagFalse

    pop dx
    pop bx
    pop ax

    ret

showNumber ENDP
```