Arquitectura de Computadores 2022-1s

Practica: Lenguaje Ensamblador

Integrantes:

1. Hinara Pastora Sanchez C.E. 1098098
2. Juan Jose Ospina Erazo C.C. 1006071024
3. Sebastian Aguinaga Velasquez C.C. 1000105467
4. Luzarait Canas Quintero C.C. 1000290584

Para esta práctica se inicia definiendo las variables necesarias para realizar los cálculos y preguntándole al usuario por el conjunto de valores de x, $y1$ y $y2$. Estas variables se definen en el bloque de $.data$ para que ensamblador reconozca que todavía no se le está indicando ninguna instrucción, simplemente se está definiendo las variables que se usarán, al igual que las variables que guardarán todos los mensajes de bienvenida.

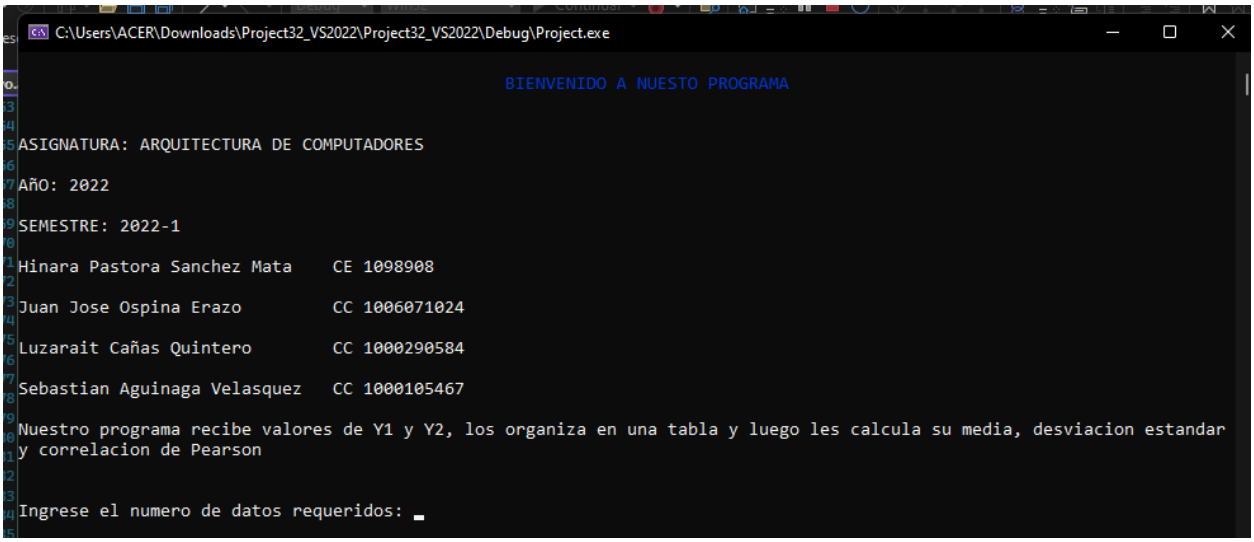
```
.data
;DEFINICIÓN DE VARIABLES
;COMENTARIOS
printdesv byte "  DESVIACION   ",0
printmed byte "     MEDIA     ",0
printper byte "    PEARSON    ",0
tetr byte "         ",0
espac byte " ",0
encab byte " ||        X        ||       Y1        ||       Y2        ||",0
rayitas  byte " ------------------------------------------------------------",0
Bienvenida0 BYTE 10, 6 dup(9),"BIENVENIDO A NUESTO PROGRAMA",10,10,13,0
Bienvenida1 byte "ASIGNATURA: ARQUITECTURA DE COMPUTADORES", 10,10,13,0
Bienvenida2 byte "A",164,"O: 2022",10,10,13,0
Bienvenida3 byte "SEMESTRE: 2022-1",10,10,13,0
Bienvenida4 byte "Hinara Pastora Sanchez Mata    CE 1098908",10,10,13,0
Bienvenida5 byte "Juan Jose Ospina Erazo         CC 1006071024",10,10,13,0
Bienvenida6 byte "Luzarait Ca",164,"as Quintero        CC 1000290584",10,10,13,0
Bienvenida7 byte "Sebastian Aguinaga Velasquez   CC 1000105467",10,10,13,0
Explicacion byte "Nuestro programa recibe valores de Y1 y Y2, los organiza en una tabla y luego les calcula su media, desviacion estandar y correlacion de Pearson",0
Mensaje1 byte "Ingrese el numero de datos requeridos: ",0
MensajeY1 byte "Ingrese los valores de Y1: ",0
MensajeY2 byte "Ingrese los valores de Y2: ",0
final Byte 10, 6 dup(9),"!GRACIAS!, VUELVA PRONTO",10,10,13,0
sep byte " || ",0
```

```
;ARRAYS
arraysito REAL8 10 DUP(?)
arraysitoY1 REAL8 10 DUP(?)
arraysitoY2 REAL8 10 DUP(?)

;VARIABLES
r Real4 ?
multi Real4 ?
multi2 Real4 ?
n1 Real4 1.0
n2 Real4 ?
x Real4 1.0
cont1 real4 0.0
cont2 real4 0.0
media real4 ?
base1 real4 ?
base2 real4 ?
potencia1 real4 0.0
potencia2 real4 0.0
desv1 real4 0.0
desv2 real4 0.0
resultado1 real4 0.0
resultado2 real4 0.0
resultado3 real4 0.0
pear real4 0.0
sumpp real4 0.0
divi real4 0.0
rest real4 0.0
```

El código empieza mostrando un mensaje de bienvenida, que previamente se guardó en .data. de la siguiente forma:

```
.code
main proc
    ;MENSAJES DE BIENVENIDA
    mov edx, offset Bienvenida0             ;guarda en edx el mensaje de bienvenida inicial
    mov eax,blue                            ;guarda en eax el color azul que sera utilizada para configurar el color del texto
    call SetTextColor                       ;se hace un llamado que establece el color del texto
    call writestring                        ;escribe el mensaje de bienvenida inicial
    call crlf                               ;Llama la función crlf para añadir un salto de línea en pantala
    mov eax,white                           ;guarda en eax el color blanco que sera utilizada para configurar el color del texto
    call SetTextColor                       ;se hace un llamado que establece el color del texto
    mov edx, offset Bienvenida1             ;Mueve a el registro edx la dirección de memoria de Bienvenida1
    call writestring                        ;Escribe en pantalla el comentario Bienvenida1
    mov edx, offset Bienvenida2             ;Mueve a el registro edx la dirección de memoria de Bienvenida2
    call writestring                        ;Escribe en pantalla el comentario Bienvenida2
    mov edx, offset Bienvenida3             ;Mueve a el registro edx la dirección de memoria de Bienvenida3
    call writestring                        ;Escribe en pantalla el comentario Bienvenida3
    mov edx, offset Bienvenida4             ;Mueve a el registro edx la dirección de memoria de Bienvenida4
    call writestring                        ;Escribe en pantalla el comentario Bienvenida4
    mov edx, offset Bienvenida5             ;Mueve a el registro edx la dirección de memoria de Bienvenida5
    call writestring                        ;Escribe en pantalla el comentario Bienvenida5
    mov edx, offset Bienvenida6             ;Mueve a el registro edx la dirección de memoria de Bienvenida6
    call writestring                        ;Escribe en pantalla el comentario Bienvenida6
    mov edx, offset Bienvenida7             ;Mueve a el registro edx la dirección de memoria de Bienvenida7
    call writestring                        ;Escribe en pantalla el comentario Bienvenida7
    mov edx, offset Explicacion             ;Mueve a el registro edx la dirección de memoria de Explicacion
    call writestring                        ;Escribe en pantalla el comentario Explicacion
    call crlf                               ;Llama la función crlf para añadir un salto de línea en pantalla
    call crlf                               ;Llama la función crlf para añadir un salto de línea en pantalla
    call crlf                               ;Llama la función crlf para añadir un salto de línea en pantalla

    ;MENSAJE DE INGRESO DATOS
    mov edx, offset Mensaje1                ;Mueve al registro edx la dirección de memoria de Mensaje1
    call writestring                        ;Escribe en pantalla el comentario Mensaje1
    call readdec                            ;Guarda en eax el valor obtenido en consola
```

Imprime lo siguiente:



A continuación, se quiere crear arrays con los valores de x, $y1$ y $y2$. Para esto primero se tiene que mover el valor de lo que hay en el registro eax tanto al registro ecx como a la variable r, esto último para usarla en otras operaciones. Luego como se quiere que el registro ebx quede en ceros (porque es la posición inicial) se usa la función XOR. Esta función es útil porque al utilizar un XOR entre la misma variable se obtiene el resultado de 0.

```
;CONTADOR A PARTIR DEL #DE DATOS
mov r, eax                          ;Guarda en r el valor de eax
mov ecx, r                          ;Guarda en ecx el valor de r, contador del siguiente loop
xor ebx, ebx                        ;Convierte a ebx en 0
datitosx:                           ;Inicia el ciclo
    fld x                           ;Carga a x en la pila
    fstp arraysito[ebx]             ;Guarda x en la posición ebx del array arraysito
    add ebx, 08h                    ;Suma 8 a ebx en hexadecimal para apuntar al indice de posicion inicial de cada elemento del array
    fld x                           ;Cargo a x en la pila
    fadd n1                         ;A x le suma n1=1
    fstp x                          ;Guarda el valor anterior en x
    finit                           ;Vacía la pila
loop datitosx                       ;Decrementa ecx y termina el ciclo cuando llegue a 0
```

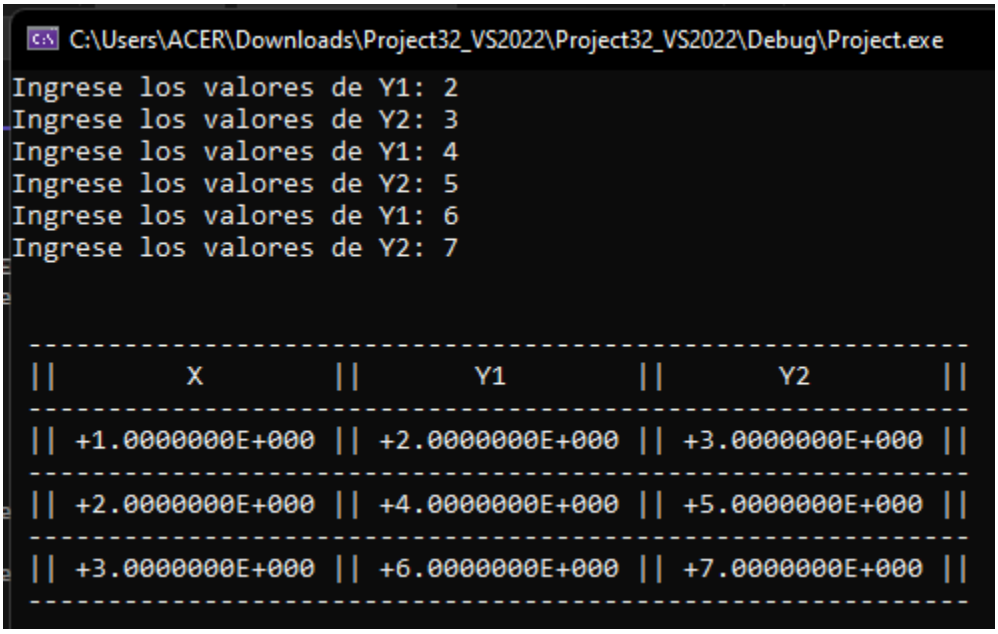
Como en nuestro código se usaron los valores de x como índices, no se necesita de la consola para obtener dichos valores, para ello se usó a x que comienza en 1 y mediante un loop que tiene como contador a $ecx = n$, se fue incrementando dicho valor en una unidad y se fue almacenando en la posición correspondiente del array.

```
;OBTENCIÓN DE Y1 Y Y2 A PARTR DE CONSOLA
xor ebx, ebx                        ;Convierte a ebx en 0
mov ecx, r                          ;Guarda en ecx el valor de r, contador del siguiente loop
datos:                              ;Inicia el ciclo
    mov edx, offset MensajeY1       ;Guarda en edx el MensajeY1
    call writestring                ;Escribe en pantalla el mensaje anterior
    call ReadFloat                  ;Lee un float de la consola
    fstp arraysitoY1[ebx]           ;Guarda en la posición ebx de arraysitoY1 el valor obtenido de consola
    mov edx, offset MensajeY2       ;Guarda en edx el MensajeY2
    call writestring                ;Escribe en pantalla el mensaje anterior
    call ReadFloat                  ;Lee un float de la consola
    fstp arraysitoY2[ebx]           ;Guarda en la posición ebx de arraysitoY2 el valor obtenido de consola
    add ebx,8                       ;Sumo 8 a ebx para apuntar al indice de posicion inicial de cada elemento del array
    finit                           ;Vacía la pila
loop datos                          ;Decrementa ecx y termina el ciclo cuando llegue a 0
```

Para obtener los valores de $y1$ y $y2$ se hizo mediante un loop que obtiene datos de la consola, dicho loop se itera con el contador $ecx = n$, muestra el mensaje de ingreso de datos en la consola y cada valor ingresado lo guarda en la posición correspondiente de cada uno de los arrays, al final suma una posición y vuelve iterar hasta que ecx llegue a 0.

```
;IMPRESIÓN DE DATOS
call crlf                           ;Añade salto de línea
call crlf                           ;Añade salto de línea
mov edx, offset rayitas             ;Mueve a el registro edx la dirección de memoria de rayitas
call writestring                    ;Escribe en pantalla el comentario rayitas
call crlf                           ;Añade un salto línea
mov edx, offset sep                 ;Mueve a el registro edx la dirección de memoria de sep
mov edx, offset encab               ;Guarda en edx el comentario encab
call writestring                    ;Muestra en pantalla el anterior comentario
call crlf                           ;Añade salto de línea
mov edx, offset rayitas             ;Mueve a el registro edx la dirección de memoria de rayitas
call writestring                    ;Escribe en pantalla el comentario rayitas
call crlf                           ;Añade un salto línea
mov edx, offset sep                 ;Mueve a el registro edx la dirección de memoria de sep
xor ebx, ebx                        ;Convierte a ebx en 0
mov ecx, r                          ;Guarda en ecx el valor de r, contador del siguiente loop
printing:                           ;Inicia el ciclo
    mov edx, offset sep             ;Guarda en edx el comentario sep
    call writestring                ;Muestra en pantalla el anterior comentario
    fld    arraysito[ebx]           ;Carga el valor del arraysito en la posición ebx en la pila
    call   WriteFloat               ;Escribe en pantalla el valor anterior
    call   writestring              ;Muestra en pantalla el comentario sep
    fld arraysitoY1[ebx]            ;Carga el valor del arraysitoY1 en la posición ebx en la pila
    call   WriteFloat               ;Escribe en pantalla el valor anterior
    call   writestring              ;Muestra en pantalla el comentario sep
    fld    arraysitoY2[ebx]         ;Carga el valor del arraysitoY2 en la posición ebx en la pila
    call   WriteFloat               ;Escribe en pantalla el valor anterior
    add ebx, 08h                    ;Suma a ebx 8 en hexadecimal para apuntar al indice de posición inicial de cada elemento del array
    call writestring                ;Muestra en pantalla el comentario sep
    call crlf                       ;Añade un salto de línea
    mov edx, offset rayitas         ;Mueve a el registro edx la dirección de memoria de rayitas
    call writestring                ;Escribe en pantalla el comentario rayitas
    call crlf                       ;Añade un salto línea
    mov edx, offset sep             ;Mueve a el registro edx la dirección de memoria de sep
    finit                           ;Vacía la pila
loop printing                       ;Decrementa ecx y termina el ciclo cuando llegue a 0
call crlf                           ;Añade un salto de línea
mov edx, offset rayitas             ;Mueve a el registro edx la dirección de memoria de rayitas
call writestring                    ;Escribe en pantalla el comentario rayitas
call crlf                           ;Añade un salto línea
mov edx, offset sep                 ;Mueve a el registro edx la dirección de memoria de sep
call writestring                    ;Muestra en pantalla el comentario sep
mov edx, offset printmed            ;Guarda en edx el comentario printmed
call writestring                    ;Muestra en pantalla el comentario printmed
mov edx, offset sep                 ;Guarda en edx el comentario sep
call writestring                    ;Muestra en pantalla el comentario sep
```

Para mostrar el array con todos los valores de x, $y1$ y $y2$ se creó un ciclo en donde se lee cada valor, se muestra en pantalla y se le agregan los mensajes de diseño correspondientes para que luzcan como en una tabla, se imprime lo siguiente:



- La Media

Teniendo en cuenta que la media de un conjunto de datos es el promedio, se utiliza la siguiente formula:

$$Media = \frac{x_1 + x_2 + x_3 + \ldots + x_n}{n}$$

Donde n es la cantidad de datos que se tiene (en nuestro caso n **no** puede tomar valores mayores a 10) y x_n son los valores correspondientes de (en nuestro caso) $y1$ o $y2$.

Por ejemplo, si se estuviera calculando la media a los valores $y1$ y el usuario ingresa 5 datos con los valores 1, 2, 3, 4 y 5, la media seria:

$$Media = \frac{y_1 + y_2 + y_3 + y_4 + y_5}{5}$$

$$Media = \frac{1 + 2 + 3 + 4 + 5}{5} = \frac{15}{5} = 5$$

Para dicho cálculo se empezó con el contador $ecx = n$ y se realizó un loop en el que se suma cada uno de los datos de cada array y al finalizar el loop el resultado obtenido se divide entre el número de datos que es igual a n, este resultado de guarda en $cont1$ para $y1$y $cont2$ para $y2$.

```
;CALCULO DE MEDIAS
;MEDIA DE Y1
xor ebx, ebx                        ;Convierte a ebx en 0
mov ecx, r                          ;Guarda en ecx el valor de r, contador del siguiente loop
suma:                               ;Inicia el loop
    fld arraysitoY1[ebx]            ;Carga el valor del arraysitoY1 en la posición ebx en la pila
    fadd cont1                      ;Suma al valor anterior cont1 que inicialmente es 0
    fstp cont1                      ;Guarda el resultado anterior en cont1
    add ebx, 08h                    ;Suma 8 a ebx en hexadecimal para apuntar al indice de posicion inicial de cada elemento del array
loop suma                           ;Decrementa ecx y termina el ciclo cuando llegue a 0
fld cont1                           ;Carga a la pila cont1
fild r                              ;Carga a la pila r
fdiv                                ;Divide cont1 entre r
call writefloat                     ;Imprime en pantalla el valor obtenido
fstp cont1                          ;Guarda el resultado de la division en cont1
```

```
;MEDIA DE Y2
xor ebx, ebx                        ;Convierte a ebx en 0
mov ecx, r                          ;Guarda en ecx el valor de r, contador del siguiente loop
suma2:                              ;Inicia el loop
    fld arraysitoY2[ebx]            ;Carga el valor del arraysitoY2 en la posición ebx en la pila
    fadd cont2                      ;Suma al valor anterior cont2 que inicialmente es 0
    fstp cont2                      ;Guarda el resultado anterior en cont2
    add ebx, 08h                    ;Suma 8 a ebx en hexadecimal para apuntar al indice de posicion inicial de cada elemento del array
loop suma2                          ;Decrementa ecx y termina el ciclo cuando llegue a 0
fld cont2                           ;Carga a la pila cont1
fild r                              ;Carga a la pila r
fdiv                                ;Divide cont1 entre r
call writestring                    ;Muestra en pantalla el comentario sep
call writefloat                     ;Imprime en pantalla el valor obtenido
fstp cont2                          ;Guarda el resultado de la division en cont2
call writestring                    ;Muestra en pantalla el comentario sep
call  crlf                          ;Añade un salto de linea
finit                               ;Vacía la pila
call writestring                    ;Muestra en pantalla el comentario sep
```

Por último se muestra en pantalla lo siguiente:

```
------------------------------------------------------------
||      MEDIA      || +4.0000000E+000 || +5.0000000E+000 ||
------------------------------------------------------------
```

- La Desviación Estándar

La desviación estándar es la que mide la dispersión de unos datos. Se puede decir que la desviación estándar es mayor entre más dispersos estén los datos. Es importante tener en cuenta que esta no puede ser negativa, sin embargo, si toma un valor cercano a cero significa que los datos tienden a estar cercanos a la media. Por otro lado, entre más lejos estén los datos de la media, mayor va a ser la desviación estándar.

Para calcular la desviación se usó la fórmula:

$$Des.Estandar = \sqrt{\frac{\sum|x - \bar{x}|^2}{N}}$$

En donde $\bar{x}$ es el valor de la media, $|x - \bar{x}|^2$ son las desviaciones (la distancia de cada dato a la media) elevado al cuadrado y N es el número de datos.

```
;DESVIACION DE Y2
xor ebx, ebx                        ;Convierte a ebx en 0
mov ecx, r                          ;Guarda en ecx el valor de r, contador del siguiente loop
desviacionY2:                       ;Inicia el loop
    fld arraysitoY2[ebx]            ;Carga el valor del arraysitoY2 en la posición ebx en la pila
    fld cont2                       ;Carga a la pila cont2
    fsub                            ;Resta ambos valores
    fstp base2                      ;Guarda el resultado de la resta en base2
    fld base2                       ;Carga a la pila base2
    fmul base2                      ;Multiplica base2 por sí misma
    fstp potencia2                  ;Guarda el resultado de la multiplicación en potencia2
    fld desv2                       ;Carga a la pila desv2
    fadd potencia2                  ;Suma potencia2 a desv2
    fstp desv2                      ;Guarda el resultado de la suma en desv2
    xor eax, eax                    ;Convierte a eax en 0
    mov potencia2, eax              ;Guarda en potencia2 el valor de eax
    add ebx, 08h                    ;Suma 8 a ebx en hexadecimal para apuntar al índice de posición inicial de cada elemento del array
    finit                           ;Vacía la pila
loop desviacionY2                   ;Decrementa ecx y termina el ciclo cuando llegue a 0
    fld desv2                       ;Carga a la pila desv2
    fild r                          ;Carga a la pila r
    fdiv                            ;Divide desv2 entre r
    fstp desv2                      ;Guarda el resultado de la división en desv2
    fld desv2                       ;Carga a la pila desv2
    fsqrt                           ;Calcula la raíz cuadrada de desv2
    call writefloat                 ;Muestra en pantalla el valor de desv2
    call writestring                ;Muestra en pantalla el comentario sep
    fstp resultado2                 ;Guarda el resultado de la raíz cuadrada en resultado2
finit                               ;Vacía la pila
call crlf                           ;Añade un salto línea
call writestring                    ;Muestra en pantalla el comentario sep
mov edx, offset printper            ;Guarda en edx el comentario printper
call writestring                    ;Muestra en pantalla el comentario printper
mov edx, offset sep                 ;Guarda en edx el comentario sep
call writestring                    ;Muestra en pantalla el comentario sep
```

El cálculo en el programa se hizo mediante el uso del contador $ecx = n$, luego a cada valor del array se le restó la media correspondiente, este valor se guardó en $base1$ para $y1$ y $base2$ para $y2$, luego se multiplicó cada $base$ por sí mismo para obtener la potencia a la 2 respectiva, y se guardaron los resultados en $potencia1$ y $potencia2$, cada valor se sumó a $desv1$ y $desv2$ para poder realizar la sumatoria total de cada resultado obtenido en cada iteración del loop. Al finalizar se muestra en pantalla los correspondientes valores de las desviaciones de $y1$ y $y2$ de la siguiente forma:

```
-------------------------------------------------------------
||   DESVIACION    || +1.6329931E+000 || +1.6329931E+000 ||
-------------------------------------------------------------
```

- La Correlación de Pearson

La correlación de Pearson mide la relación entre dos variables en donde el resultado puede tomar valores entre -1 y 1. En el caso de que la correlación de menor a cero, se puede decir que se presenta una correlación negativa por lo que las variables estarían relacionadas inversamente (mientras aumente el valor de una variable, el valor de la otra disminuye). Si la correlación toma un valor de 0 significa que no hay asociación entre las variables. Por otro lado, si da un valor mayor a cero, esto indica una relación proporcional, o sea que mientras aumente el valor de una variable, también aumentará el valor de la otra variable.

Si se quiere calcular la correlación de Pearson se debe usar la fórmula:

$$r_{xy} = \frac{\sum z_x z_y}{N}$$

En donde x e y son las variables a las cuales se les quiere calcular el coeficiente de Pearson, z_x es la desviación estándar de la variable x, z_y es la desviación estándar de la variable y, y N es el numero de datos que tenemos.

La fórmula también se puede calcular de la siguiente manera:

$$r_{xy} = \frac{\frac{\sum XY}{N} - \bar{X}\bar{Y}}{S_x S_y}$$

Esta última fórmula fue la que se usó en nuestro programa, de la siguiente forma:

```
;SUMATORIA DE X*Y
xor ebx, ebx                    ;Convierte a ebx en 0
mov ecx, r                      ;Guarda en ecx el valor de r, contador del siguiente loop
sump:                           ;Inicia el loop
    fld arraysitoY1[ebx]        ;Carga el valor del arraysitoY1 en la posición ebx en la pila
    fld arraysitoY2[ebx]        ;Carga el valor del arraysitoY2 en la posición ebx en la pila
    fmul                        ;Multiplica los dos valores anteriores
    fstp sumpp                  ;Guarda el resultado de la multiplicación en sumpp
    fld pear                    ;Carga a la pila pear
    fadd sumpp                  ;Suma sumpp a pear
    fstp pear                   ;Guarda el resultado de la suma en pear
    add ebx, 08h                ;Suma 8 a ebx en hexadecimal para apuntar al índice de posición inicial de cada elemento del array
    finit                       ;Vacía la pila
loop sump                       ;Decrementa ecx y termina el ciclo cuando llegue a 0

;MULTIPLICACION MEDIAS
fld cont1                       ;Carga a la pila cont1
fmul cont2                      ;Multplica cont1 por cont2
fstp multi2                     ;Guarda el resultado de la multiplicación en multi2

;MULTIPLICACION DESVIACIONES
fld resultado1                  ;Apila el valor de resultado1
fmul resultado2                 ;Multiplica el valor de resultado1 por resultado2
fstp multi                      ;Desapila el valor de la operación anterior y lo almacena en multi
```

Primero se calculó la sumatoria de la multiplicación de cada término de $y1$ por cada término de $y2$, este resultado se guardó en la variable $pear$.

Luego se calculó la multiplicación de las medias de $y1$ y $y2$ y el resultado se guardó en la variable $multi2$, también se calculó la multiplicación de las desviaciones de $y1$ y $y2$ y el resultado obtenido se guardó en $multi$.

```
;PEARSON
fld pear                        ;Apila el valor de pear
fild r                          ;Carga el valor del entero r
fdiv                            ;Divide pear por el valor de r
fstp divi                       ;Desapila el valor de la operación anterior y lo almacena en divi
fld divi                        ;Apila el valor de divi
fsub multi2                     ;Resta multi2 al valor de la cabeza de la pila
fstp rest                       ;Desapila el valor de la operación anterior y lo almacena en rest
fld rest                        ;Apila el valor de rest
fld multi                       ;Apila el valor de multi
fdiv                            ;Divide a rest por el valor de multi
mov edx, offset tetr            ;Guarda en edx el comentario tetr
call writestring                ;Muestra en pantalla el comentario tetr
call writefloat                 ;Muestra en pantalla el valor de la división
fstp resultado3                 ;Desapila el valor de la operación anterior y lo almacena en resultado3
call writestring                ;Muestra en pantalla el comentario tetr
mov edx, offset espac           ;Guarda en el registro edx el comentario espac
call writestring                ;Muestra en pantalla el valor de espac
mov edx, offset sep             ;Guarda en el registro edx el comentario sep
call writestring                ;Muestra en pantalla el valor de sep
call crlf                       ;Llama la función crlf para añadir un salto de línea en pantala
mov edx, offset final           ;Guarda en el registro edx el comentario final
mov eax,red                     ;Mueve al registro eax el valor del color red
call SetTextColor               ;Cambia en consola el color cargado anterior en eax
call writestring                ;Muestra en pantalla el comentario  final
call crlf                       ;Llama la función crlf para añadir un salto de línea en pantala
fwait                           ;Sincroniza el procesador y el coprocesador
```

Para finalizar se hizo uso de las variables *pear*, *multi*2 y *multi* para calcular el resultado de la correlación de Pearson, para ello primero se apiló *pear* y se dividió entre r que es el número de datos N, luego a este resultado se le restó el valor de *multi*2 y por último lo obtenido se dividió entre *multi* y así se obtuvo finalmente el valor de la correlación de Pearson. Para concluir se imprime en pantalla el valor y el mensaje final de despedida de la siguiente forma:

```
------------------------------------------------------------
||     PEARSON     ||          +9.9999973E-001           ||
------------------------------------------------------------
```

```
                    !GRACIAS!, VUELVA PRONTO
```