

Benemérita Universidad Autónoma de Puebla  
Facultad de Ciencias de la Computación

Benemérita Universidad Autónoma de Puebla  
Facultad de Ciencias de la Computación

# SISTEMAS OPERATIVOS I

---

## PRACTICA 4: INTERFAZ C-ASM



Docente:

Prof. Marcos González Flores

Alumno:

Jesús Huerta Aguilar

Alex Abdiel Ruano Flores

Matricula:

202041509

202075025

NRC: 46152

Sección: 003

## QUINTO SEMESTRE

Puebla, Pue.

23/02/2022

Nombre: Interfaz C-ASM

Objetivo: Aprender a realizar una interfaz entre 2 lenguajes diferentes, en particular C – ASM.  
También se aprenderá a generar ejecutables.

Desarrollo:

## PRACTICA 1

```
holaxd.c

#include <stdio.h>

int main(){
    extern int imprimo();
    int y;
    printf("programa que hace el llamado de la funcion imprime\n");
    imprimo();
}
```

```
imp.asm

global imprimo
extern printf
section .data
    msg db "hola mundo!!", eah,
section .text
tmprimo:
    push dword msg
    call printf
    add esp,4
    ret
```

## TRANSCRITO + EJECUCIÓN:



```
GNU nano 6.2 holaxd.c *
#include <stdio.h>

int main(){
    extern int imprimo();
    int y;
    printf("programa quehace el llamado de la funcion imprime\n");
    imprimo();
}
```

[ 9 líneas leídas ]

^G Ayuda	^O Guardar	^W Buscar	^K Cortar	^T Ejecutar	^C Ubicación
^X Salir	^R Leer fich.	^E Reemplazar	^U Pegar	^J Justificar	^_ Ir a línea

```
GNU nano 6.2                                imp.asm
global imprimo
extern printf
section .data
    msg : db "hola mundo!!", 0ah, 0
section .text
imprimo:
    push dword msg
    call printf
    add esp,4
    ret
```

[ 10 líneas leídas ]

^G Ayuda	^O Guardar	^W Buscar	^K Cortar	^T Ejecutar	^C Ubicación
^X Salir	^R Leer fich.	^Y Reemplazar	^U Pegar	^J Justificar	^_ Ir a línea

```
alumno@alumno-VirtualBox: ~
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$ nasm -f elf32 imp.asm
alumno@alumno-VirtualBox:~$ gcc -m32 -o saludo holaxd.c imp.o
/usr/bin/ld: imp.o: warning: relocation in read-only section `.text'
/usr/bin/ld: warning: creating DT_TEXTREL in a PIE
alumno@alumno-VirtualBox:~$ ./saludo
programa quehace el llamado de la funcion imprime
hola mundo!!
alumno@alumno-VirtualBox:~$
```

## EJERCICIO 2

```
variable.c

#include <stdio.h>

int extern suma(int v1,int v2);

int main() {
    int v1,v2, sum;
    printf("Dame un dato para sumar\n");
    scanf("%d", &v1);
    printf("Dame otro numero para sumar\n");
    scanf("%d", &v2);
    sum = suma(v1,v2);
    printf("El resultado de la suma es = %d",sum);
    return 0;
}
```

```
pasoparam.asm

global suma;

suma:
    push dword ebp
    mov ebp, esp;
    mov eax, [ebp + 8]
    mov edx, [ebp+12]
    add eax, edx
    pop dword ebp
    ret
```

## TRANSCRITO + EJECUCIÓN:




```
GNU nano 6.2 variable.c
#include <stdio.h>

int extern suma(int v1,int v2);

int main(){
    int v1,v2,sum;
    printf("Dame un dato para sumar\n");
    scanf("%d",&v1);
    printf("Dame otro numero para sumar\n");
    scanf("%d",&v2);
    sum = suma(v1,v2);
    printf("El resultado de la suma es = %d",sum);
    return 0;
}
```

[ 14 líneas leídas ]

^G Ayuda ^O Guardar ^W Buscar ^K Cortar ^T Ejecutar ^C Ubicación  
^X Salir ^R Leer fich. ^E Reemplazar ^U Pegar ^J Justificar ^\_ Ir a línea

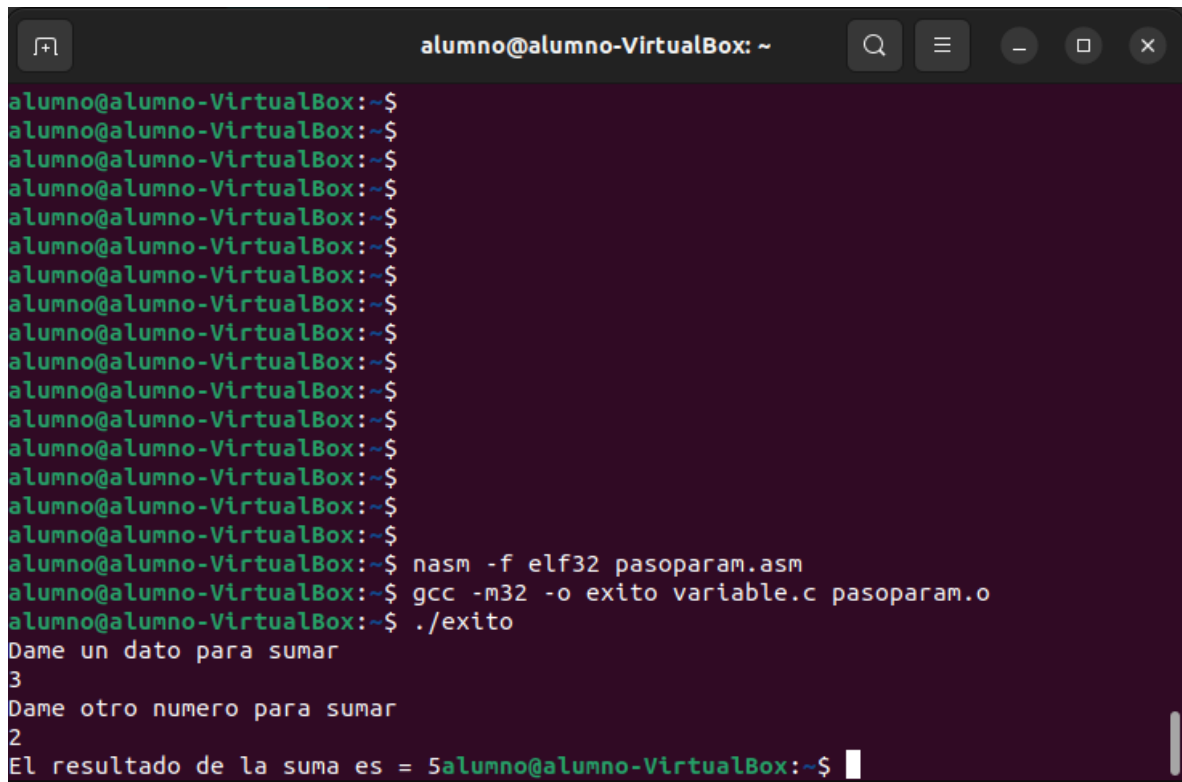


```
GNU nano 6.2 pasoparam.asm
global suma;

suma:
    push dword ebp
    mov ebp, esp;
    mov eax, [ebp + 8]
    mov edx, [ebp+12]
    add eax,edx
    pop dword ebp
    ret
```

[ 10 líneas leídas ]

^G Ayuda    ^O Guardar    ^W Buscar    ^K Cortar    ^T Ejecutar    ^C Ubicación  
^X Salir    ^R Leer fich. ^\_ Reemplazar ^U Pegar    ^J Justificar ^/ Ir a línea



```
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$
alumno@alumno-VirtualBox:~$ nasm -f elf32 pasoparam.asm
alumno@alumno-VirtualBox:~$ gcc -m32 -o exito variable.o pasoparam.o
alumno@alumno-VirtualBox:~$ ./exito
Dame un dato para sumar
3
Dame otro numero para sumar
2
El resultado de la suma es = 5alumno@alumno-VirtualBox:~$
```

EJERCICIO 3:

```
                                collatz.c

#include <stdio.h>

extern int half();
int main() {
    int count,x,y;
    count=0;
    printf("Entra un numero: ");
    scanf("%d",&x);
    while(x!=1)
    {
        count=count+1;
        y=half(x);
        printf("\n devuelve %d \n",y);
        if(y!=0)
            x=y;
        else
            x=x* 3+1;
        printf("\n X=%d",x);
    }
    printf("\n hay %d iteracciones. \n\n", count);
}
```

```
                                half.asm

global half

half: push ecx
      mov ecx, [esp+8]
      mov eax, 0
agn:  inc eax
      sub ecx, 2
      jg agn
      jz dun
      mov eax, 0
dun:  pop ecx
      ret
```

TRANSCRITO + EJECUCIÓN:

```
GNU nano 6.2                                collatz.c
#include <stdio.h>
extern int half();
int main(){
    int count,x,y;
    count=0;
    printf("Entra un numero: ");
    scanf("%d",&x);

    while(x!=1)
    {
        count=count+1;
        y=half(x);
        printf("\n Devuelve %d \n",y);
        if(y!=0)
            x=y;
        else
            x=x*3+1;
        printf("\n X=%d",x);
    }
    printf("\n hay %d iteracciones. \n\n", count);
}
```

```
GNU nano 6.2                                half.asm
global half

half: push ecx
      mov ecx, [esp+8]
      mov eax,0
agn:  inc eax
      sub ecx,2
      jg agn
      jz dun
      mov eax,0
dun:  pop ecx
      ret
```

```
alumno@alumno-VirtualBox:~$ gcc -m32 -o hola2 collatz.c half.o
alumno@alumno-VirtualBox:~$ nasm -f elf32 half.asm
alumno@alumno-VirtualBox:~$ gcc -m32 -o hola2 collatz.c half.o
alumno@alumno-VirtualBox:~$ ./hola2
Entra un numero: 4

devuelve 2

X=2
devuelve 1

X=1
hay 2 iteracciones.

alumno@alumno-VirtualBox:~$ ./hola2
Entra un numero: 0

devuelve 0

X=1
hay 1 iteracciones.

alumno@alumno-VirtualBox:~$ nano collatz.c
alumno@alumno-VirtualBox:~$ ./hola2
Entra un numero: 1

hay 0 iteracciones.

alumno@alumno-VirtualBox:~$
```

```
alumno@alumno-VirtualBox:~$ nano collatz.c
alumno@alumno-VirtualBox:~$ nano half.asm
alumno@alumno-VirtualBox:~$ gcc -o collatz collatz.c
/usr/bin/ld: /tmp/ccn5t7CI.o: en la función `main':
collatz.c:(.text+0x62): referencia a `half' sin definir
collect2: error: ld returned 1 exit status
alumno@alumno-VirtualBox:~$ gcc -m32 -o hola2 collatz.c half.o
alumno@alumno-VirtualBox:~$ nasm -f elf32 half.asm
alumno@alumno-VirtualBox:~$ gcc -m32 -o hola2 collatz.c half.o
alumno@alumno-VirtualBox:~$ ./hola2
Entra un numero: 4

devuelve 2

X=2
devuelve 1

X=1
hay 2 iteracciones.

alumno@alumno-VirtualBox:~$ ./hola2
Entra un numero: 0

devuelve 0

X=1
hay 1 iteracciones.

alumno@alumno-VirtualBox:~$
```