


UNIVERSIDAD AUTÓNOMA “TOMAS FRÍAS” CARRERA DE INGENIERÍA DE SISTEMAS				
Materia:	Arquitectura de computadoras (SIS-522)			
Docente:	Ing. Gustavo A. Puita Choque			N Práctica  9
Auxiliar:	Univ. Aldrin Roger Perez Miranda			
Estudiante:	Univ. Sergio Moises Apaza Caballero			
16/06/2024	Fecha publicación			
01/07/2024	Fecha de entrega			
Grupo:	1	Sede:	Potosí	

**1) ¿Qué es el 'stack' en el contexto del lenguaje ensamblador y cómo se utiliza?**

Es un segmento especial de memoria que funciona en conjunción con varias instrucciones de assembler, este tiene tres propósitos:

- Preservar valores de los registros temporalmente.
- Almacenar direcciones a las cuales las rutinas regresarán.
- Almacenar variables dinámicas.

El significado de stack en español es “pila”, por tanto, este opera con las reglas de esta estructura de datos, es decir, sigue una estructura de LIFO (Last In First Out).

**2) Describe un escenario práctico donde el uso de ensamblador sería más ventajoso que el uso de un lenguaje de alto nivel.**

Se pueden dar distintos casos en los cuales el uso de ensamblador es más eficiente que algún lenguaje de alto nivel; la lógica en la que se basa esta respuesta es que al ser un lenguaje ensamblador, este tiene un control más preciso sobre las operaciones realizadas por el microprocesador.

Un caso importante sería la programación de drivers para dispositivos que tengan el sistema operativo basado en GNU/Linux, ya que se tendrá un control más preciso del microprocesador para la conexión y toma de decisiones de dispositivos externos, tales como impresoras, escáneres y otros.

Cabe mencionar que, si bien es sabido que es posible realizar drivers en Visual Studio C++, se tiene más control del microprocesador con lenguaje ensamblador.

**3) Explique cada línea del siguiente código del lenguaje ensamblador y diga que es lo que se está haciendo**

```
MOV AX, 5      ; Línea 1
MOV BX, 10     ; Línea 2
ADD AX, BX     ; Línea 3
MOV CX, AX     ; Línea 4
```

**Línea 1:** Mueve el valor 5 al registro AX

**Línea 2:** Mueve el valor 10 al registro BX

**Línea 3:** Suma el registro BX con el valor de AX y lo almacena en AX

**Línea 4:** Mueve el valor de AX al registro CX

El código realiza una operación de suma simple:

AX = 5

BX = 10

AX = BX + AX

CX = AX

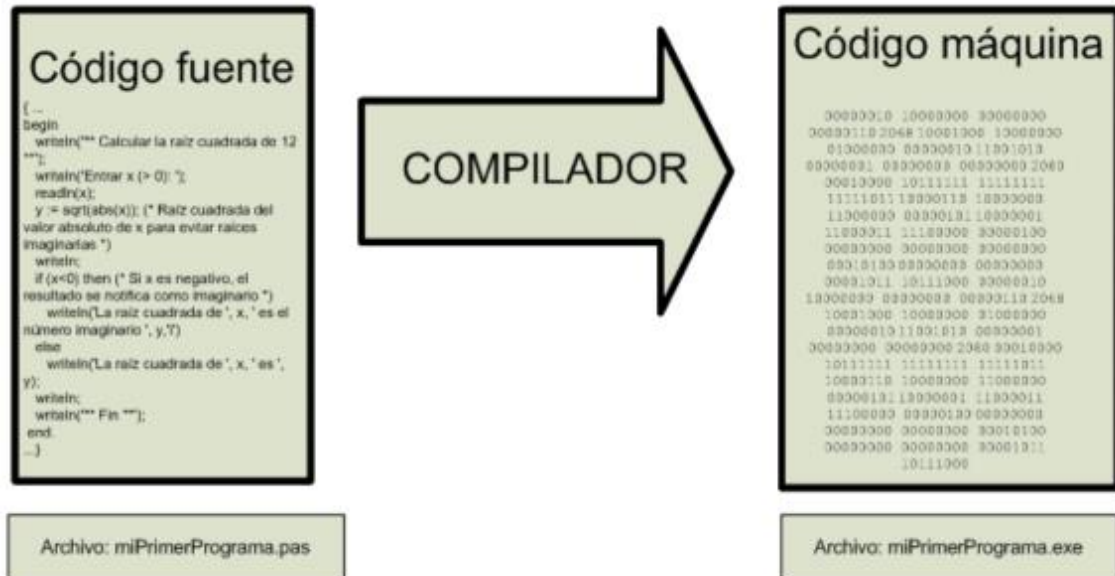
Resultados:

AX = 15

BX = 10

CX = 15

#### 4) Explique detalladamente cómo funcionan los compiladores



El compilador se encarga de convertir código fuente siguiendo el siguiente proceso:

##### 1. Análisis léxico

El compilador convierte el código en tokens, es decir, identifica el código fuente en palabras clave.

##### 2. Análisis sintético

El compilador toma estos tokens y los organiza en una secuencia lógica y entendible. Esto para verificar que se cumpla con las reglas del lenguaje utilizado.

##### 3. Análisis semántico

El compilador realiza comprobaciones de semántica del código.

##### 4. Optimización de código

Se optimiza el código con técnicas como la eliminación de código redundante, optimización de bucles, reducción del uso de memoria.

##### 5. Generación de código

Este código optimizado en el punto 4, se pasa a código máquina para que el procesador pueda ejecutarlo.

##### 6. Enlazado

El compilador combina el código generado con bibliotecas con otros módulos para tener un ejecutable completo.

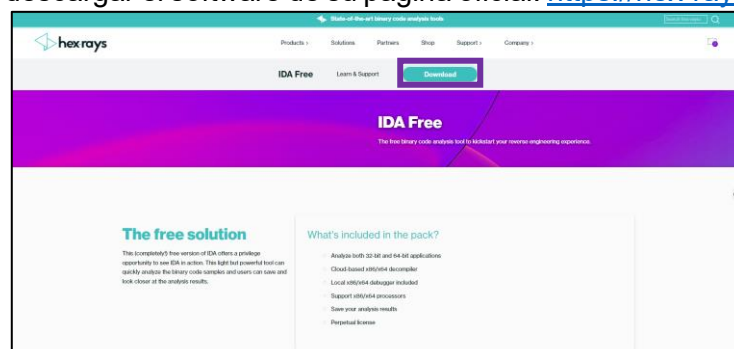
#### 5) Realizar capturas de pantalla del siguiente procedimiento:

##### Instalación de IDA

IDA es un software muy conocido software de análisis de código binario y desensamblado.

##### Paso 1: DESCARGA

1.1. Buscar y descargar el software de su página oficial: <https://hex-rays.com/ida-free/>





## 1.2. Escoger el sistema operativo (Windows en mi caso):


### Download your IDA Free


The Free version of IDA v8.3 comes with the following limitations:

- no commercial use is allowed
- cloud-based decompiler lacks certain advanced commands
- lacks support for many processors, file formats, etc...
- comes without technical support

 IDA Free for Windows (90MB)

 IDA Free for Linux (76MB)

 IDA Free for Mac (68MB)


 IDA Free for Mac ARM (70MB)

SHA256 checksums:

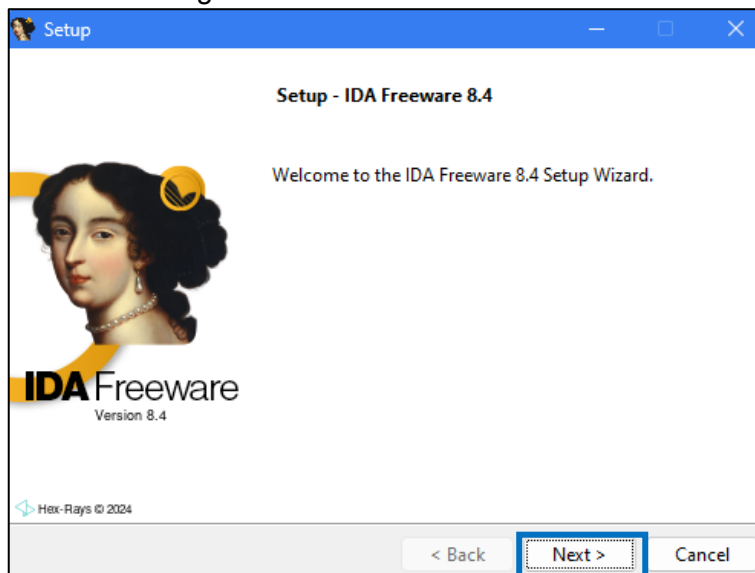
```
dec15875d871a398088a05320ac45e1971940de279a0b0e2c057054833da18fe  arm_idafree84_mac.app.zip
941f228ce4890f0a142699808eda16140be297e0749245d7839a2a3b02870a62  idafree84_linux.run
c4cf8b35082b217351ca0730c8aada70b017aac3e7421e0b420c5fe2451e6f5  idafree84_mac.app.zip
a2fc7ee918006d095c940d1ee8a059af9061e8fc5f965de411206010ac2091  idafree84_windows.exe
```

## Paso 2: INSTALACIÓN

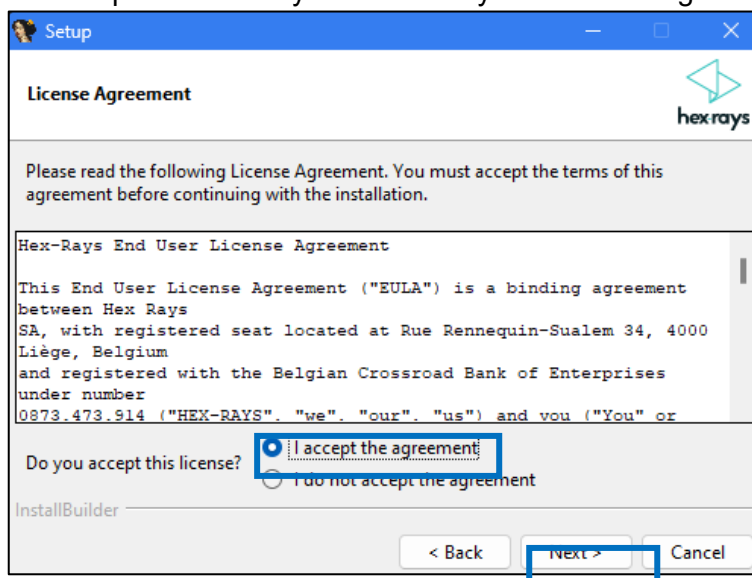
Teniendo el instalador, se procede a realizar la instalación:

 idafree84\_windows.exe

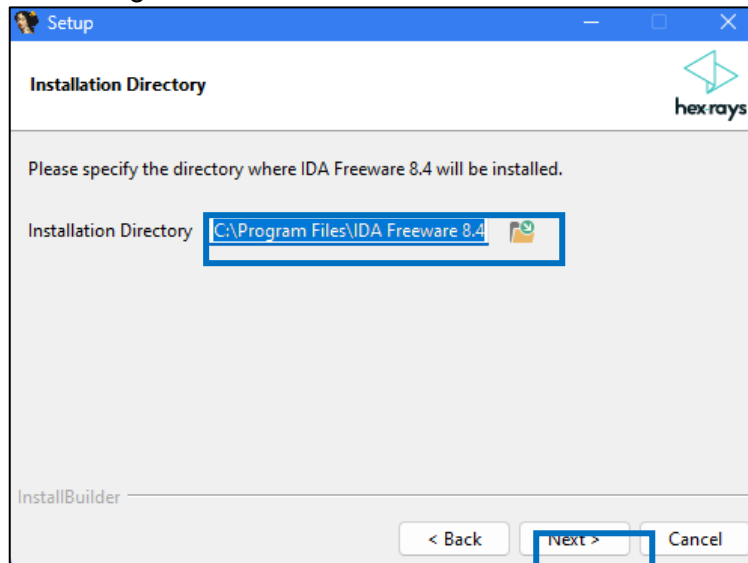
### 2.1. Click en siguiente:



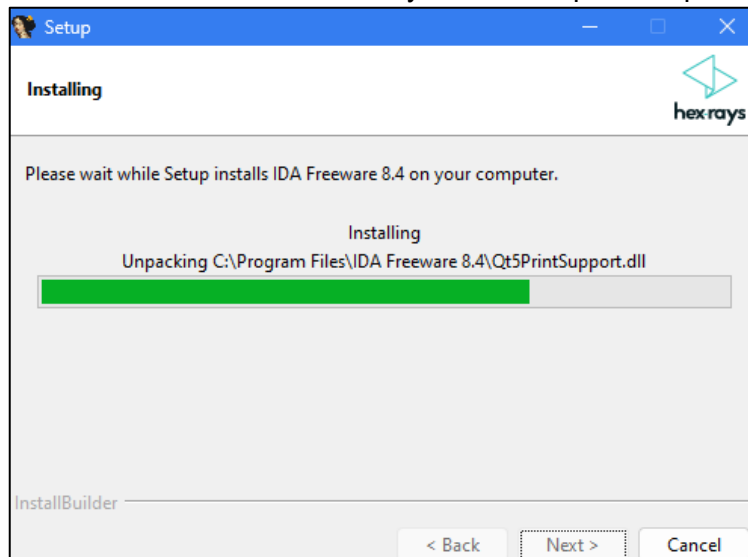
### 2.2. Aceptar términos y condiciones y dar click en siguiente:



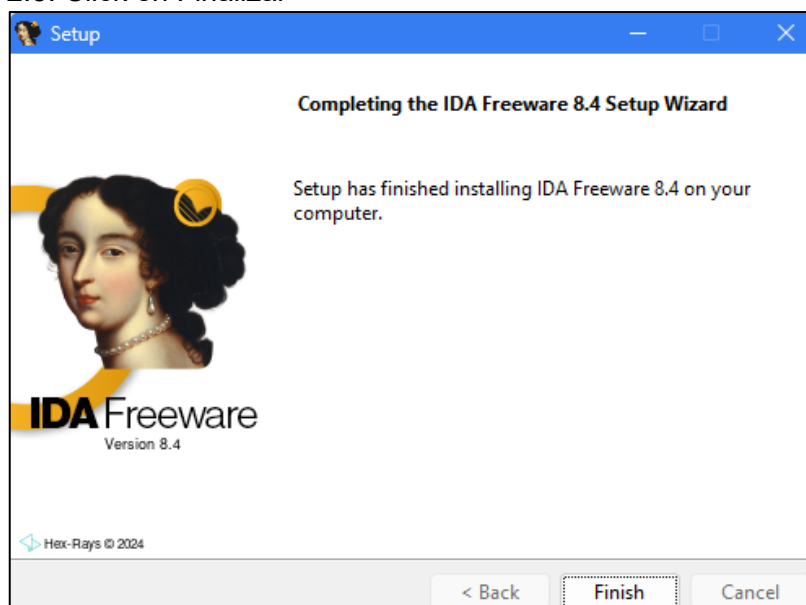
### 2.3. Escoger ruta de instalación:



### 2.4. Se confirma la instalación y se debe esperar a que esta misma concluya:



### 2.5. Click en Finalizar

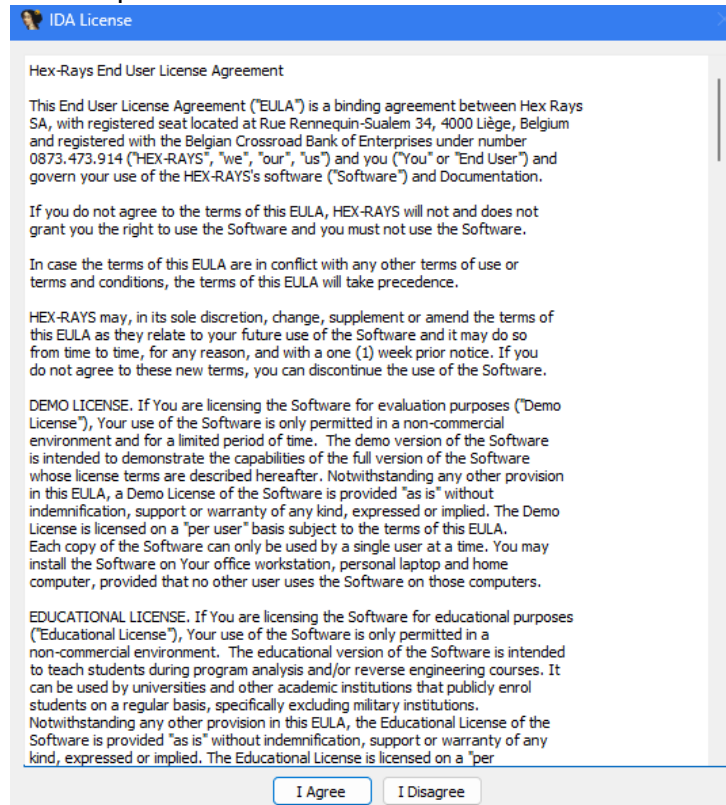


### Paso 3: UTILIZACIÓN DEL EJECUTABLE

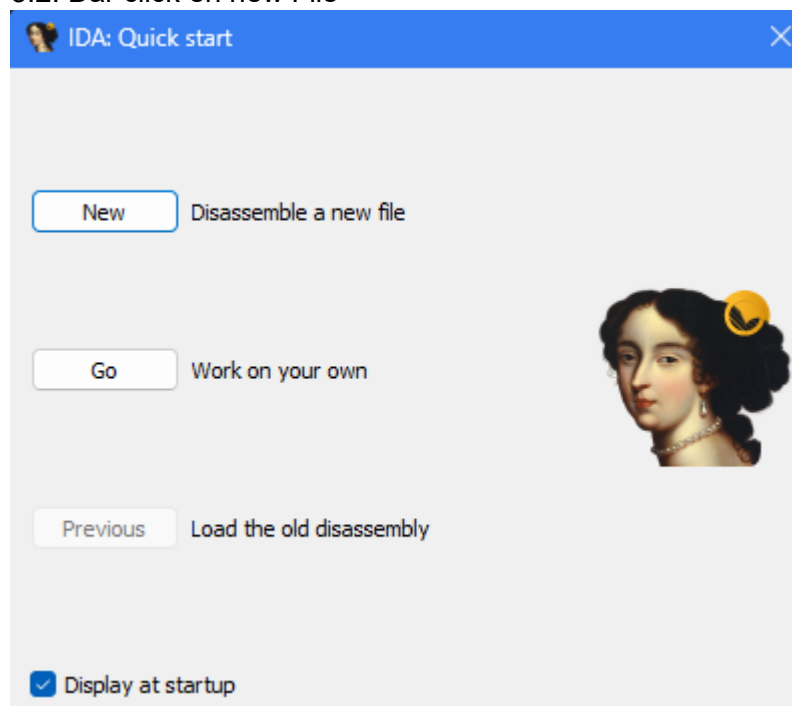


Teniendo el ejecutable, se procede a realizar lo siguiente:

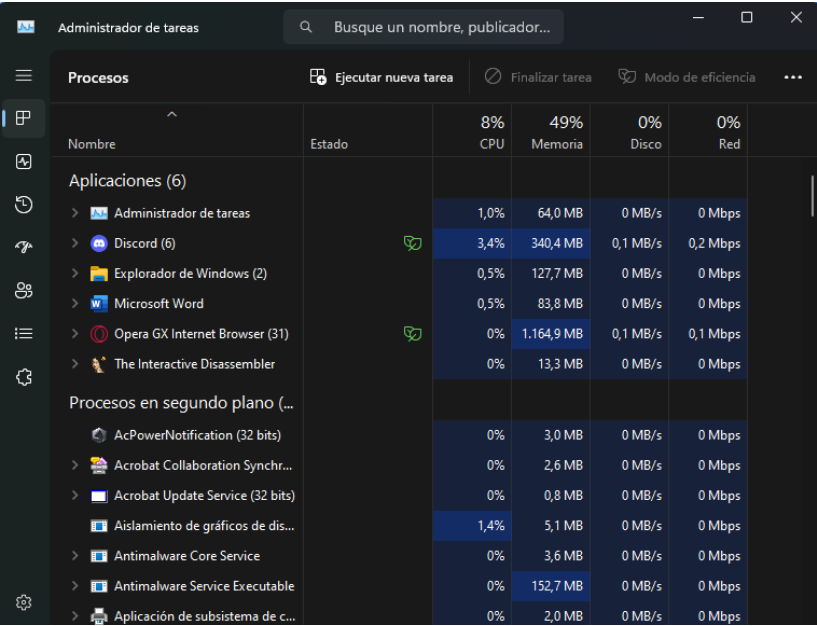
#### 3.1. Aceptar la licencia de uso



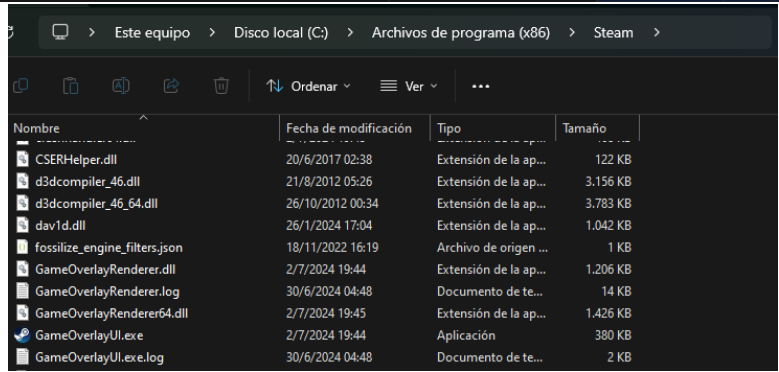
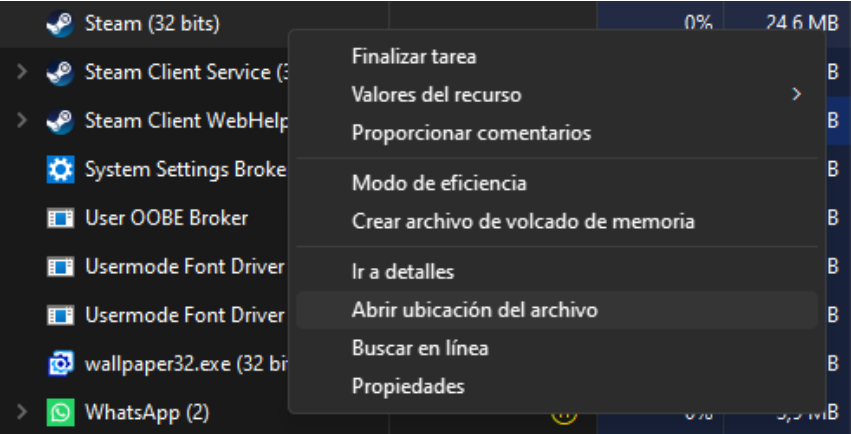
#### 3.2. Dar click en new File



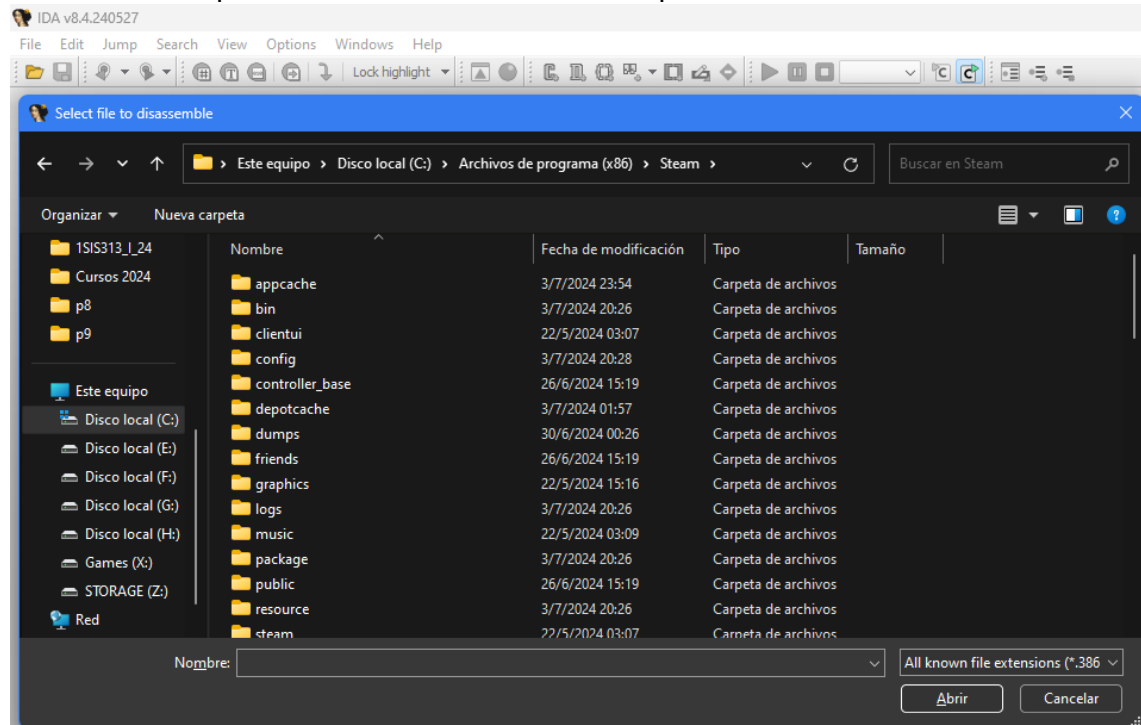
3.3. Debemos seleccionar un servicio del administrador de tareas, para ello, abrimos el administrador de tareas



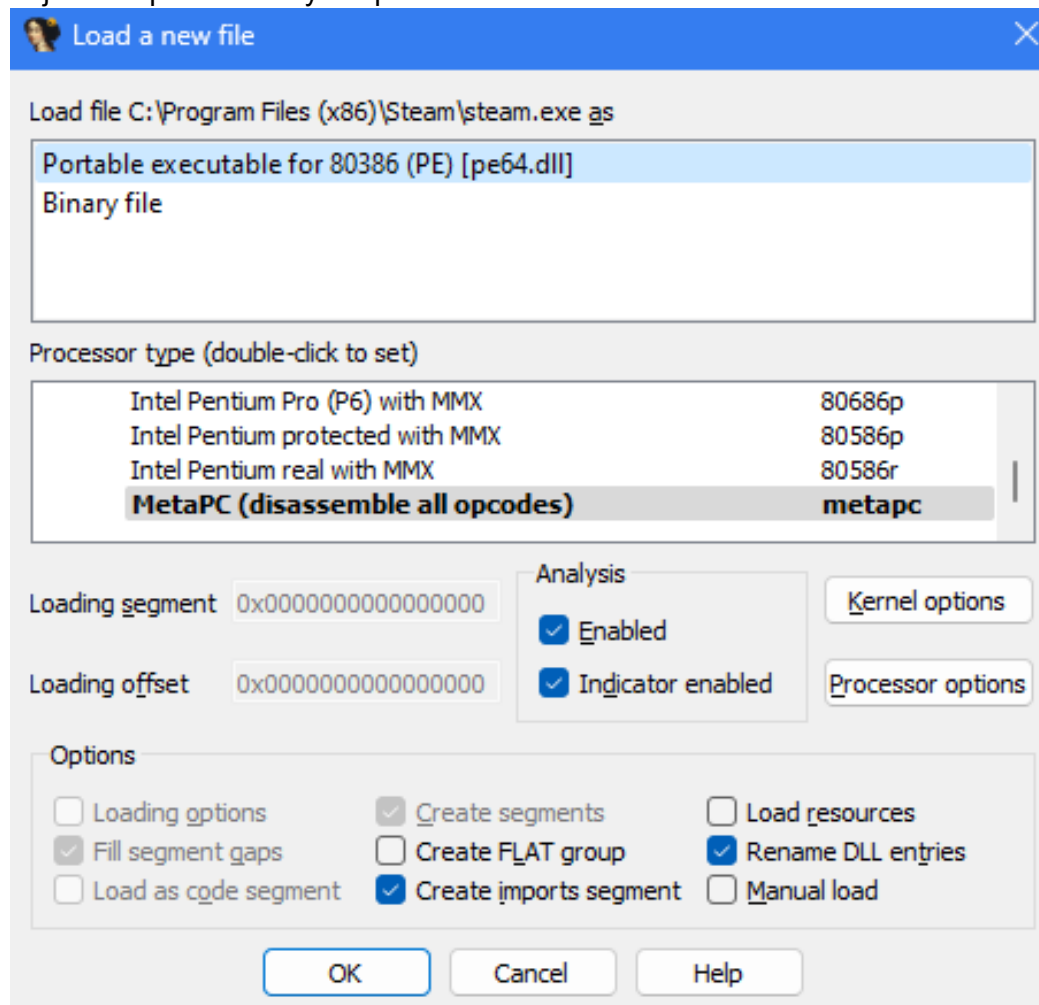
3.4. Seleccionar algún servicio, y abrir la ubicación de dicho archivo; en este caso se utilizará Steam



### 3.5. Se debe copiar la ruta del archivo donde nos pide en el software IDA



### 3.6. Al escoger el servicio (steam en este caso), nos abrirá una ventana, debemos dejar todo por defecto y aceptar



## Paso 4: DESENSAMBLAJE DEL SERVICIO

```
.text:0046CA5C      call     sub_5696F7
.text:0046CA61      mov      esi, [eax]
.text:0046CA63      call     sub_5696F1
.text:0046CA68      push     esi
.text:0046CA69      push     dword ptr [eax]
.text:0046CA6B      call     sub_456120
.text:0046CA70      add      esp, 8
.text:0046CA73      pop      esi
.text:0046CA74      pop      ebp
.text:0046CA75      retn     10h
.text:0046CA75      _WinMain@16      endp
.text:0046CA75      ; -----
.text:0046CA78      align 10h
.text:0046CA80      ; ===== S U B R O U T I N E =====
.text:0046CA80      ; Attributes: bp-based frame
.text:0046CA80      sub_46CA80      proc near          ; CODE XREF: sub_46D1A0+24C↓p
.text:0046CA80      var_10          = dword ptr -10h
.text:0046CA80      var_C           = dword ptr -0Ch
.text:0046CA80      var_4           = dword ptr -4
.text:0046CA80      arg_0           = dword ptr 8
.text:0046CA80      ✓.text:0046CA80      push     ebp
.text:0046CA81      mov      ebp, esp
.text:0046CA83      push     0FFFFFFFh
.text:0046CA85      push     offset SEH_46CA80
.text:0046CA8A      mov      eax, large fs:0
.text:0046CA90      push     eax
.text:0046CA91      mov      large fs:0, esp
.text:0046CA98      push     ecx
.text:0046CA99      push     esi
.text:0046CA9A      mov      esi, ecx
.text:0046CA9C      mov      [ebp+var_10], esi
.text:0046CA9F      mov      dword ptr [esi], 0
.text:0046CAA5      mov      [ebp+var_4], 0
.text:0046CAAC      cmp      dword ptr [esi], 0
.text:0046CAAF      jz       short loc_46CACA
.text:0046CAB1      push     offset aMPNull ; "m_p == NULL"
.text:0046CAB6      push     49h ; 'I'
.text:0046CAB8      push     offset aCBuildslaveSte ; "c:\\buildslave\\steam_rel_client_win32"...
.text:0046CABD      call     sub_539EF0
.text:0046CAC2      add      esp, 0Ch
.text:0046CAC5      test     al, al
```