

PRACTICA_09_ARQUITECTURA DE COMPUTADORAS

Nombre: Univ. Jacob Santos Ayaviri Condori

C.I. 13229452



PRACTICA ANULADA

COPIA DE ALEX MENDEZ MOREIRA

GIT HUB

Nambre: Jacob_Santos_Ayaviri_Condori

https://github.com/jacob-santos/Practica_01.git




The screenshot shows the GitHub interface for a repository named 'Practica_01'. At the top, the repository name is followed by a 'Public' badge. To the right are buttons for 'Pin' and 'Unwatch' (with a count of 1). Below this is a navigation bar with a 'main' branch selector, a 'Go to file' button, a '+' button, and a 'Code' button. The repository owner 'jacob-santos' and the name 'tarea2' are displayed, along with the commit hash 'e09a199' and the text 'last week'. A list of files follows: 'Ayaviri_Condori_Jacob_Sa...' (labeled 'POR') from '3 weeks ago', and 'Practica02_Ayaviri_Condo...' (labeled 'tarea2') from 'last week'. Below the file list is a 'README' section with a book icon and a large empty space.

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

R: el stack es una estructura de los datos en memoria que tiene el principio LIFO y se utiliza para guardar datos de forma temporal, como por ejemplo valores de registro, direcciones de retorno en llamada a funciones y parámetros







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

- el uso eficiente de la memoria: genera un código que ocupa muy poca RAM y almacenamiento 
- Accede directamente a registro y periféricos 
- Minimizar los ciclos del CPU, en especial para las tareas en tiempo real como control de motores 

Se tenía que "describir" el escenario

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

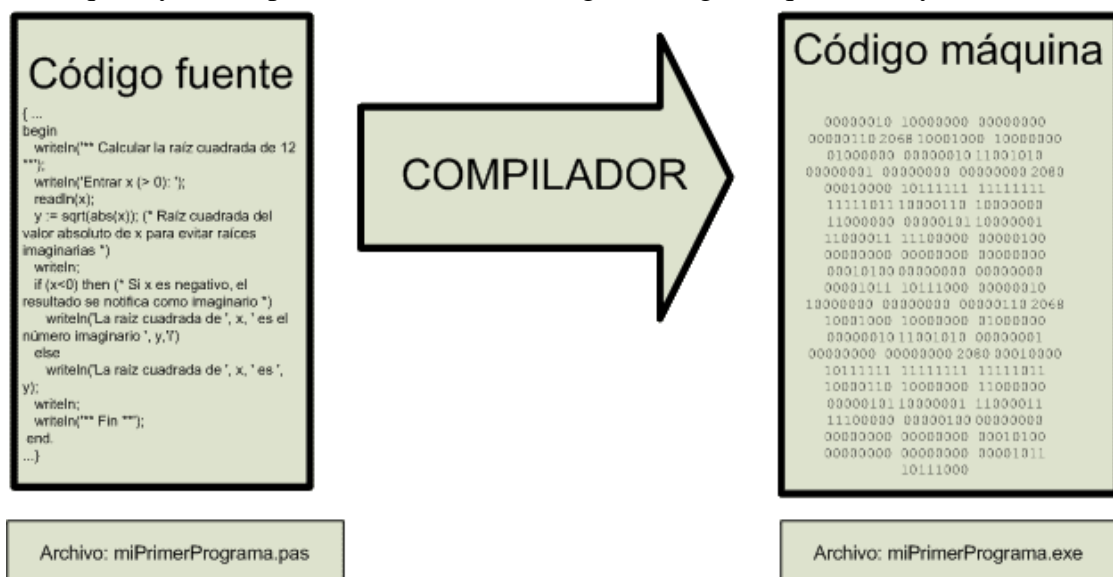
```
MOV AX, 5 ; Línea 1
MOV BX, 10 ; Línea 2
ADD AX, BX ; Línea 3
MOV CX, AX ; Línea 4
```

- Línea 1: AX = 5, el valor 5 se almacena en AX 
- Línea 2: BX=10, el valor 10 se almacena en BX 
- Línea 3: AX= 15, el add suma los dos valores y se almacena en AX 
- Línea 4: CX=15, el valor de AX se copia al CX 

4) Explique detalladamente cómo funcionan los compiladores

R: Toma el código fuente en lenguaje de alto nivel y lo traduce a lenguaje máquina, por ejemplo el código fuente es entendible para las personas pero no para la máquina y el compilador convierte el código a código máquina de 0 y 1

Debia ser una explicacion mas "detallada"



5) Realizar sus propias capturas de pantalla del siguiente procedimiento:

IDA: Es una de las herramientas más conocidas y potentes para el análisis de código binario y desensamblado. En este laboratorio se instalará IDA FREE pero también se tiene la versión de paga IDA PRO

Paso 1:

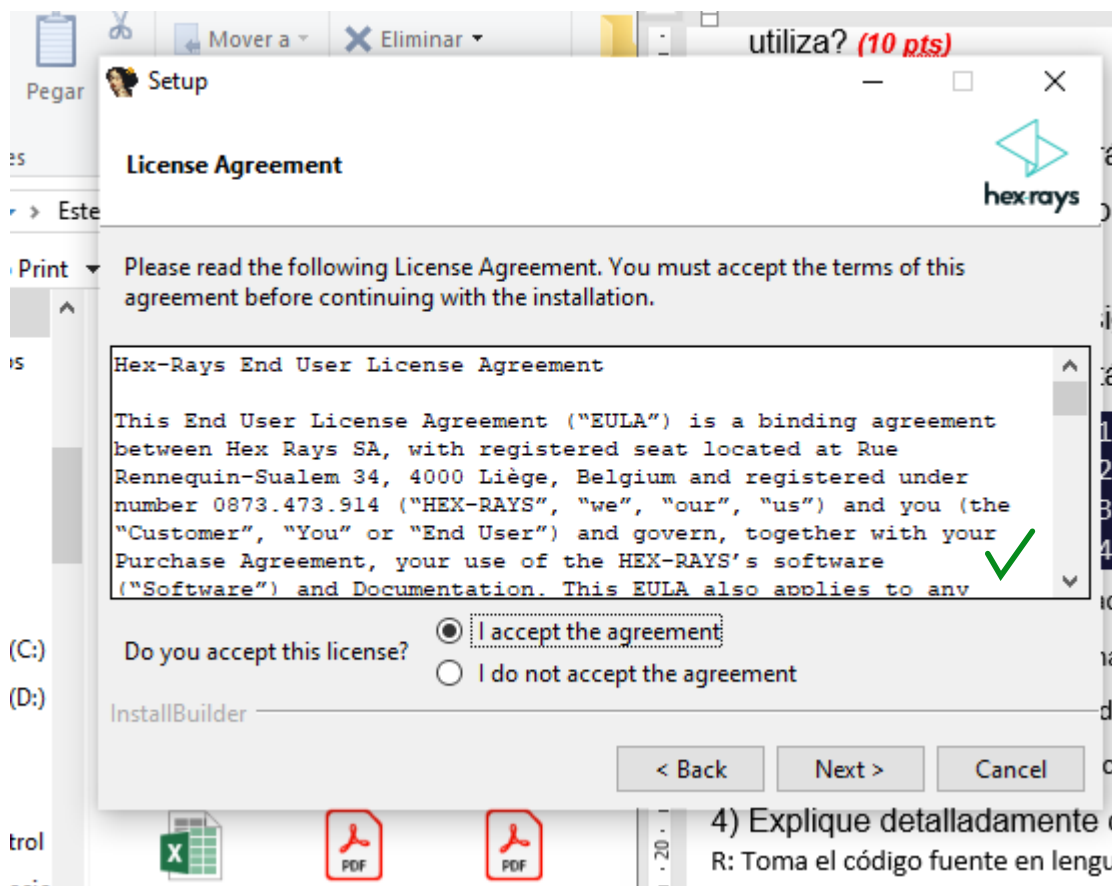
Descargar el software IDA FREE



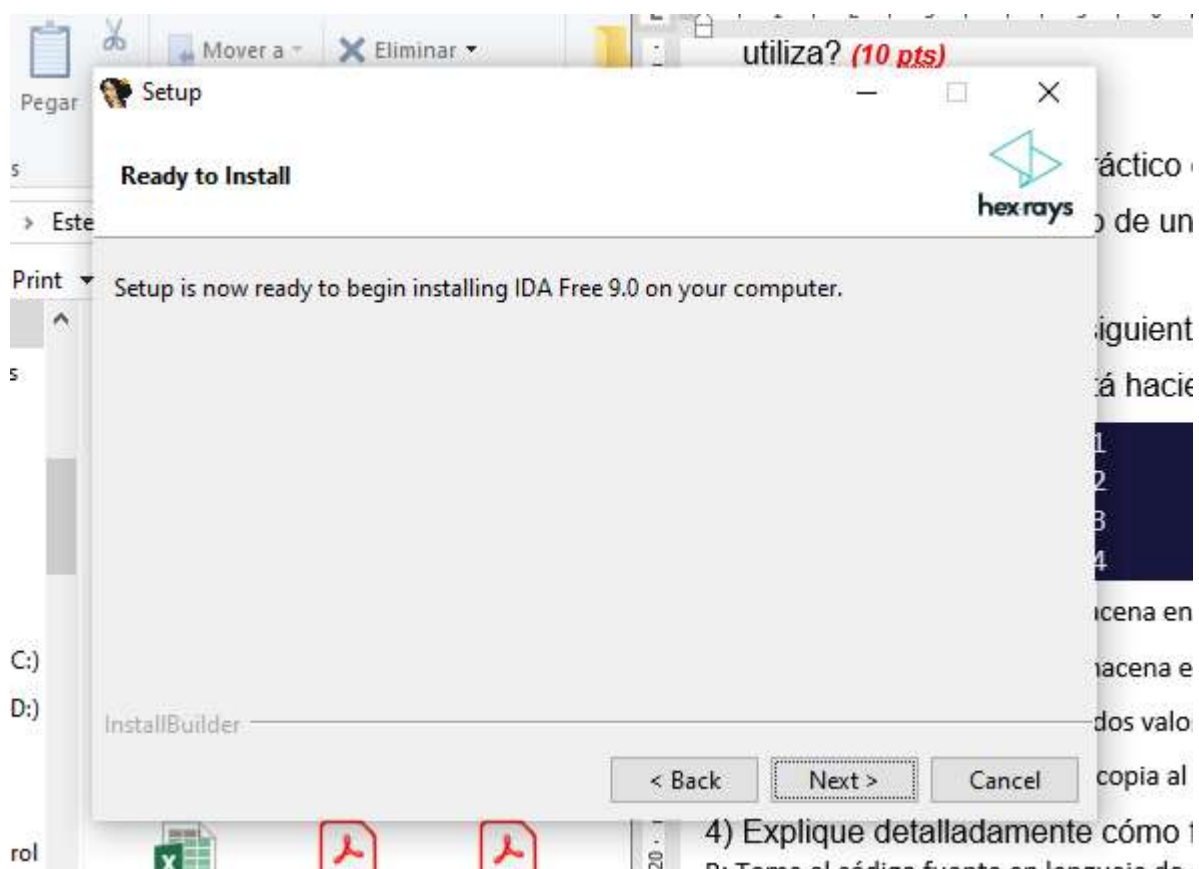
Paso 2:

Instalar el IDA FREE

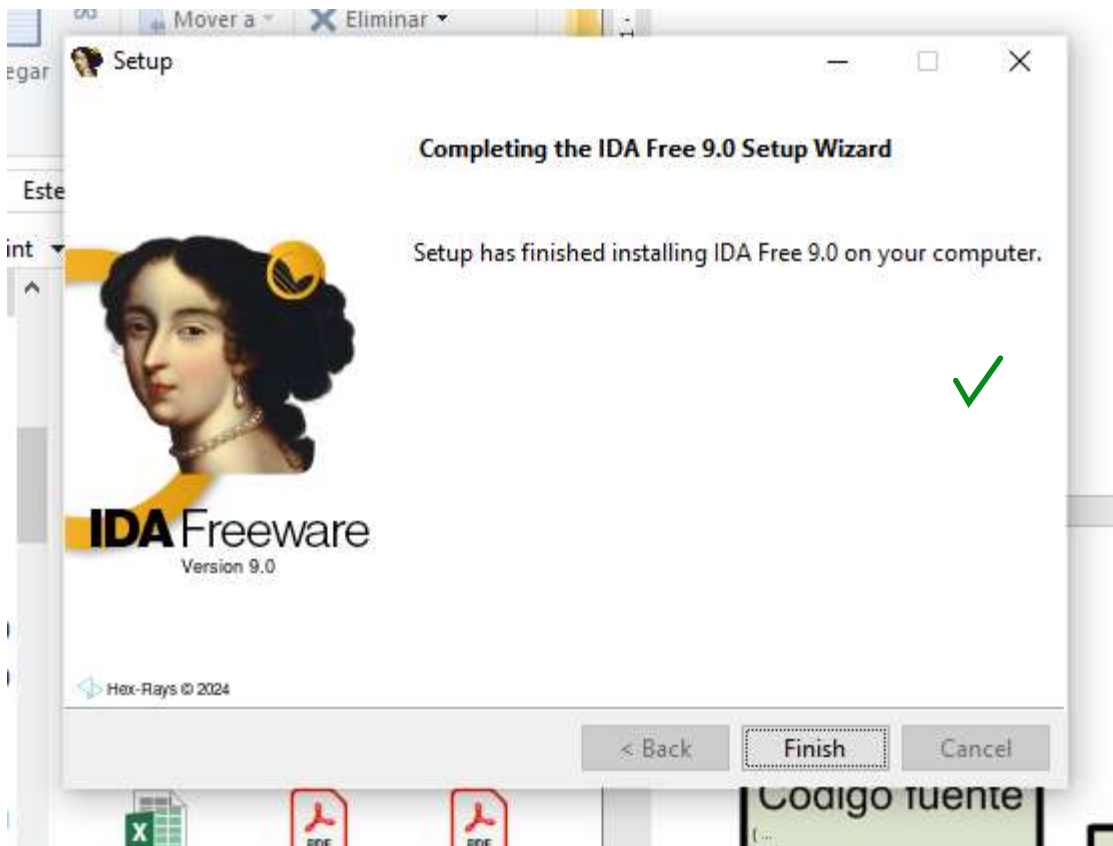




4) Explique detalladamente c
R: Toma el código fuente en lengu

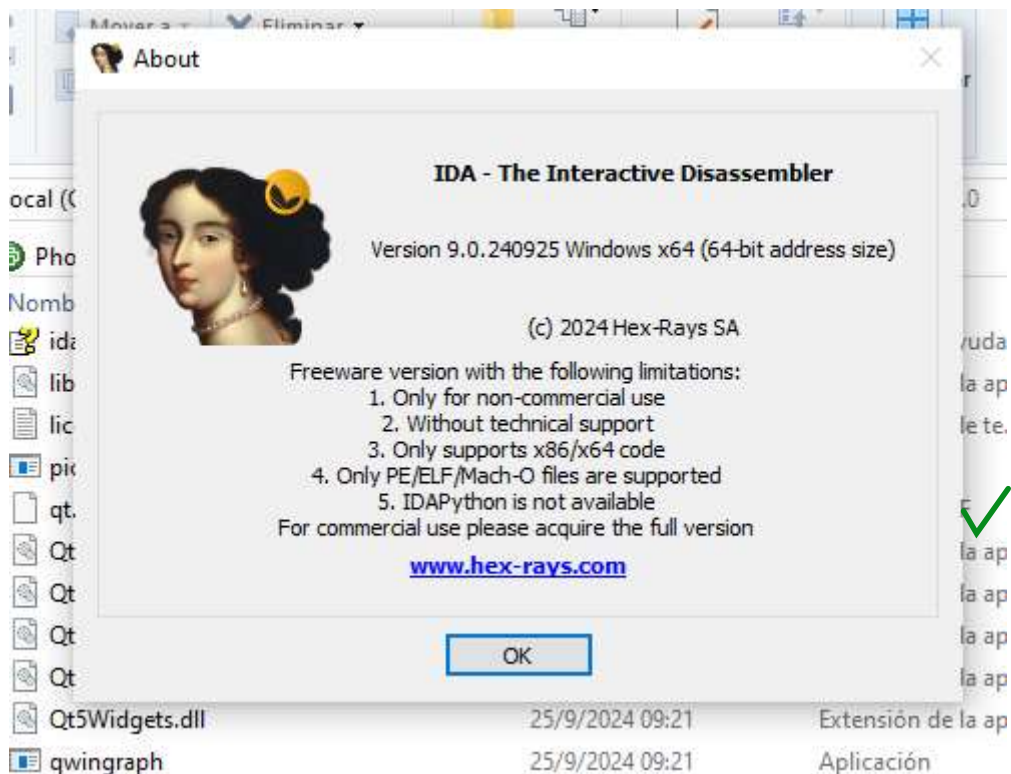


4) Explique detalladamente cómo
R: Toma el código fuente en lenguaje de

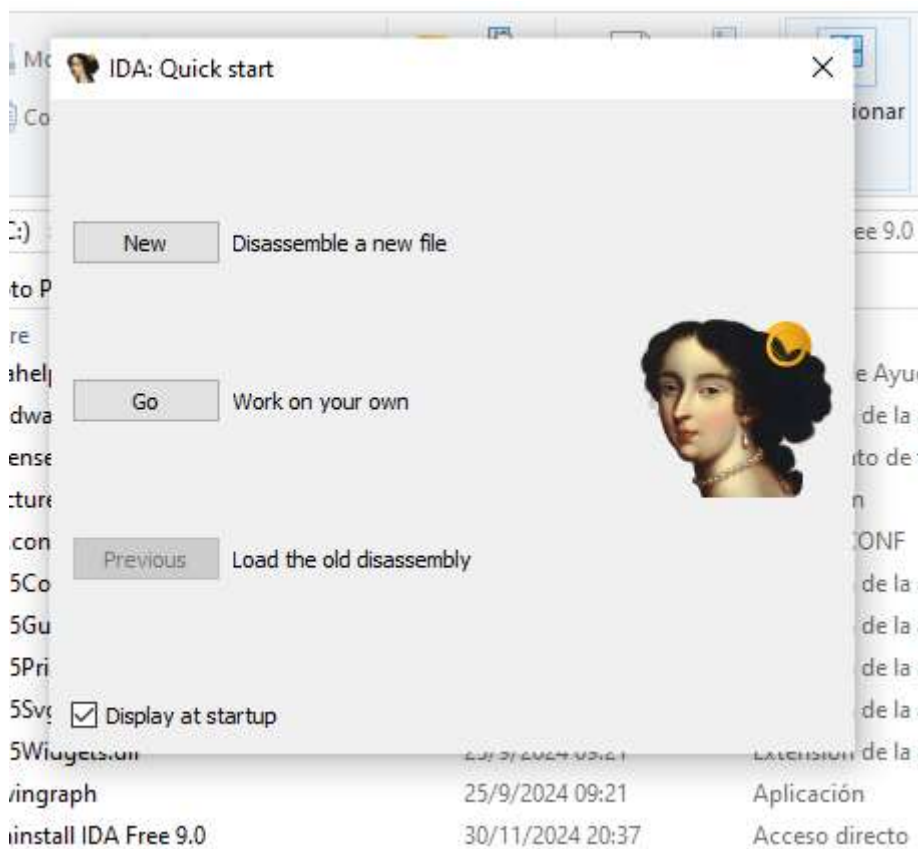


Paso 3:

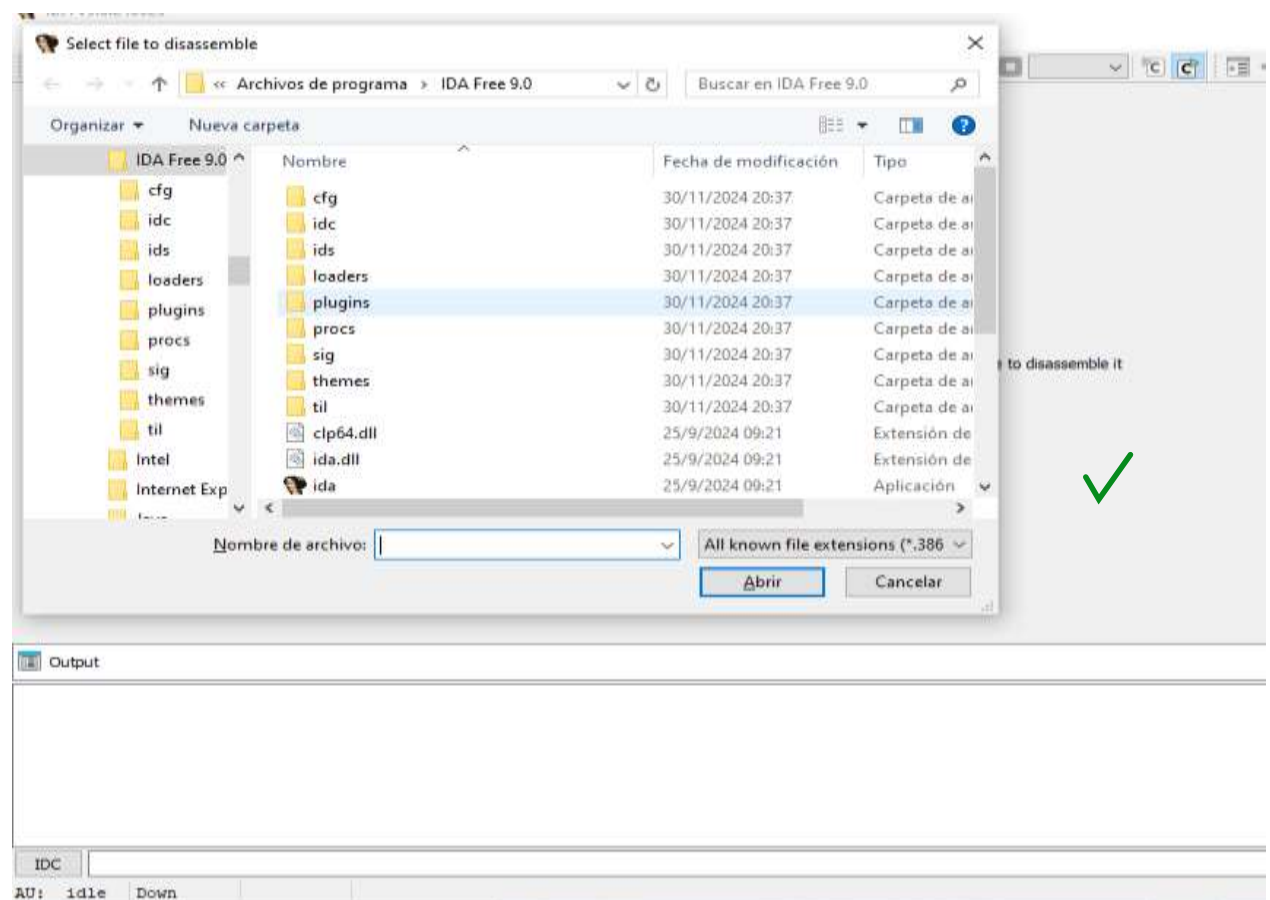
Procederemos a abrir un servicio en Windows



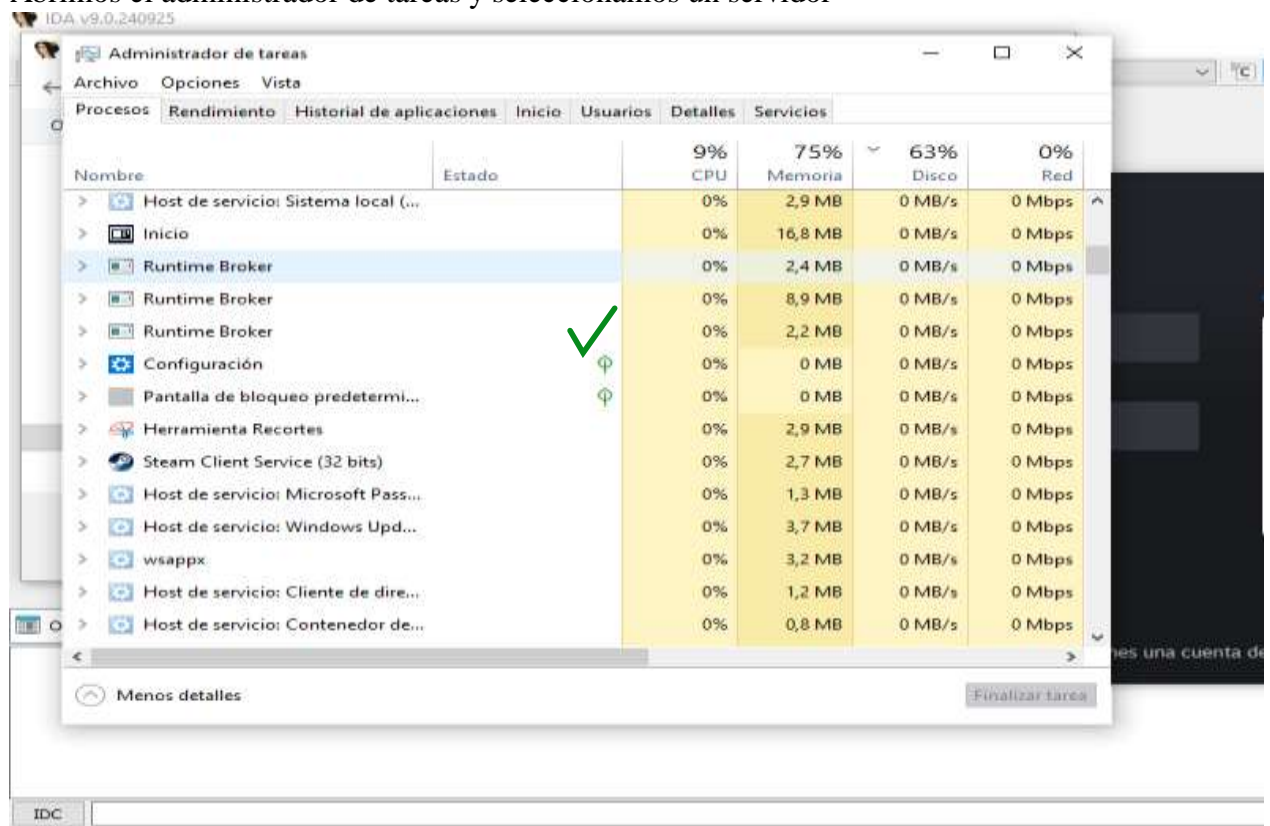
Seleccionamos new



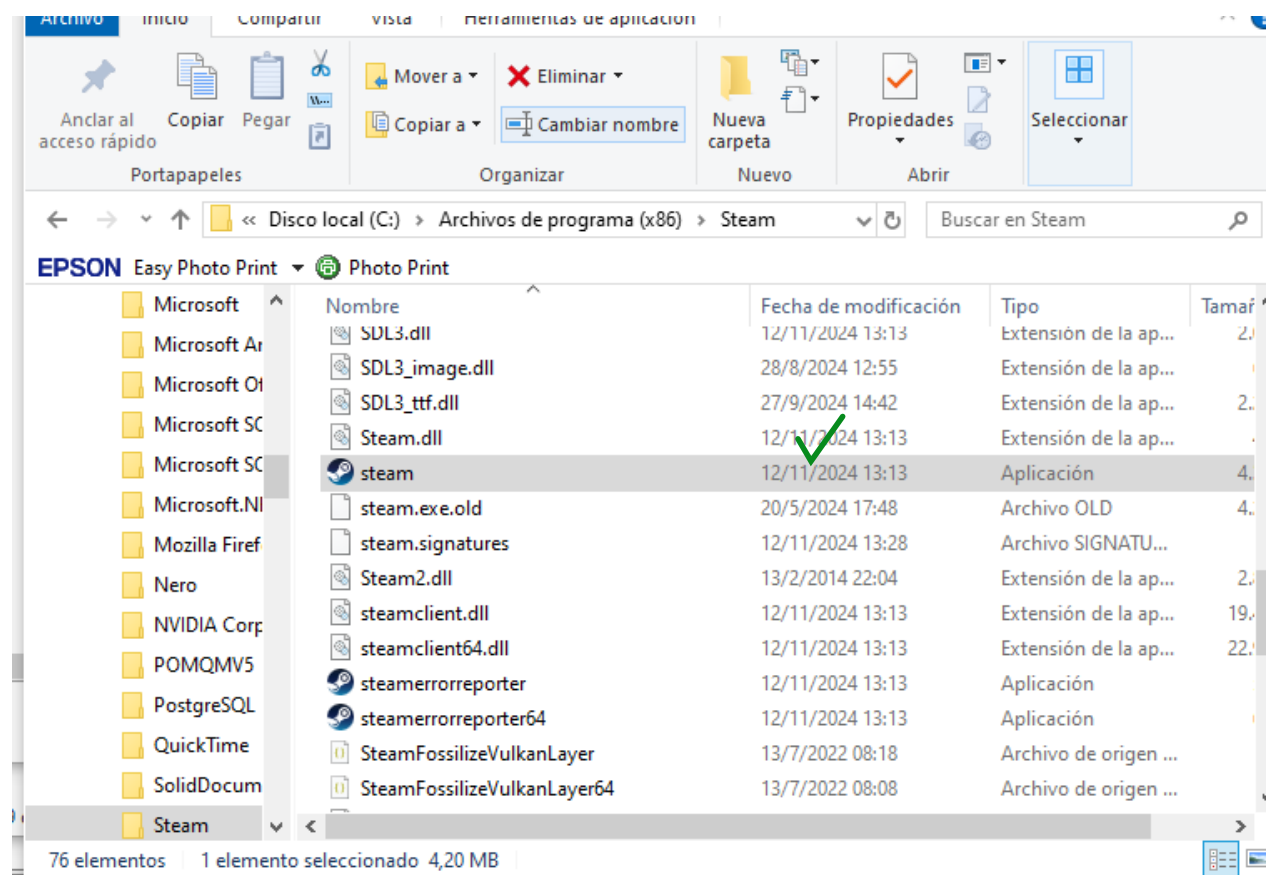
Seleccionamos un servidor en este caso steam



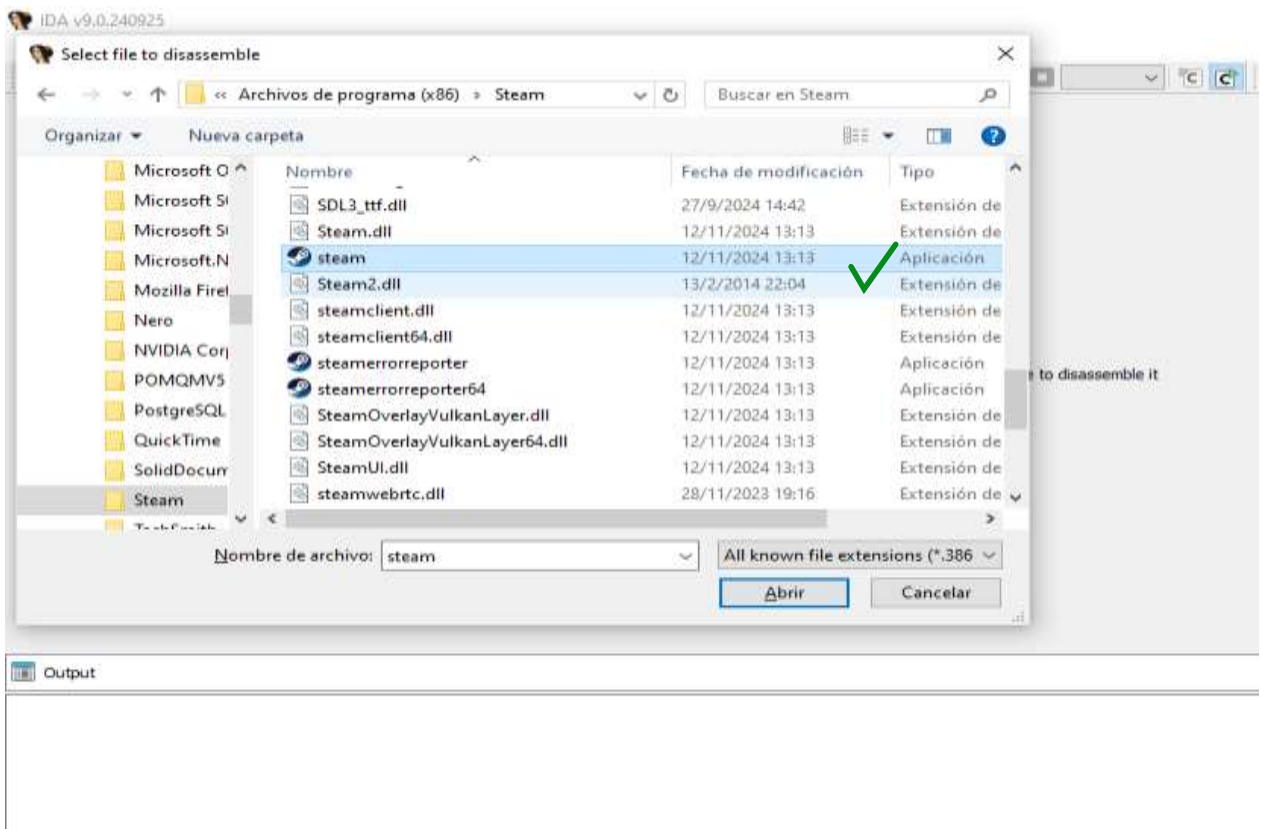
Abrimos el administrador de tareas y seleccionamos un servidor



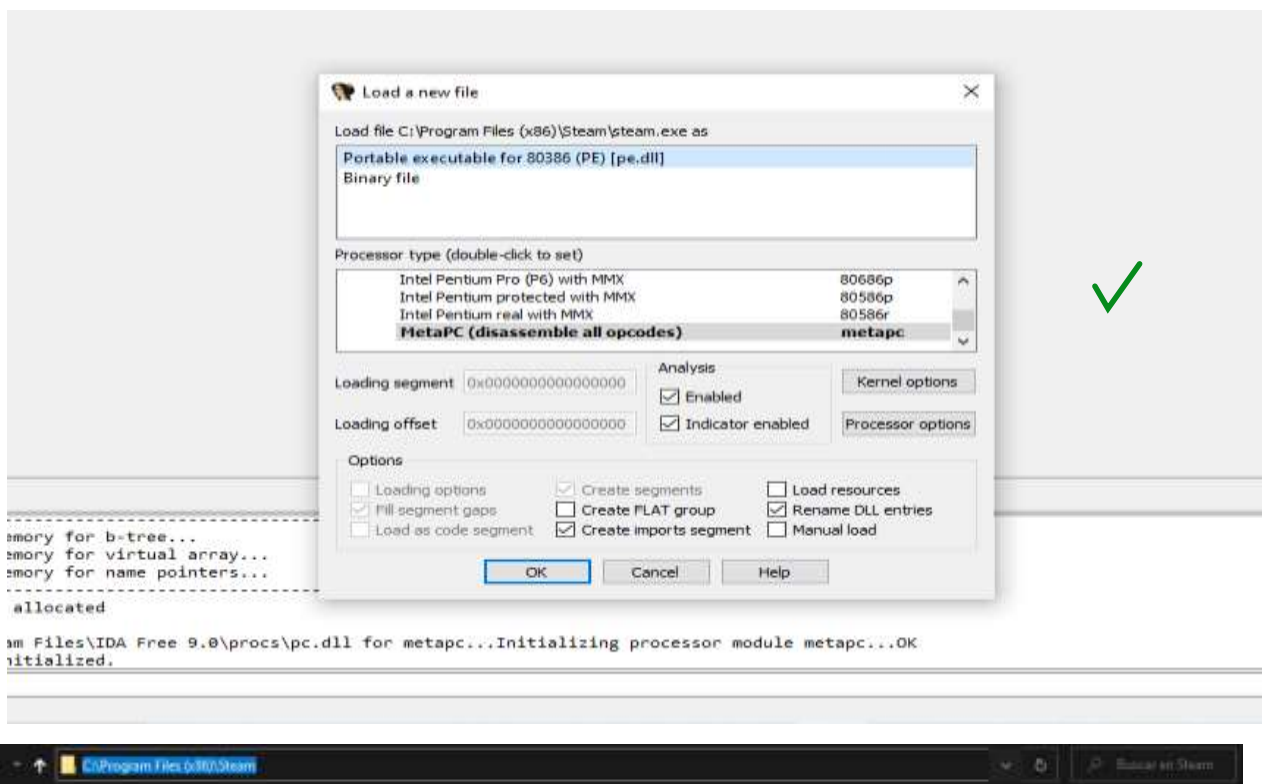
Abrimos el administrador de tareas y hacemos un clic derecho seleccionar “Abrir ubicación del archivo”



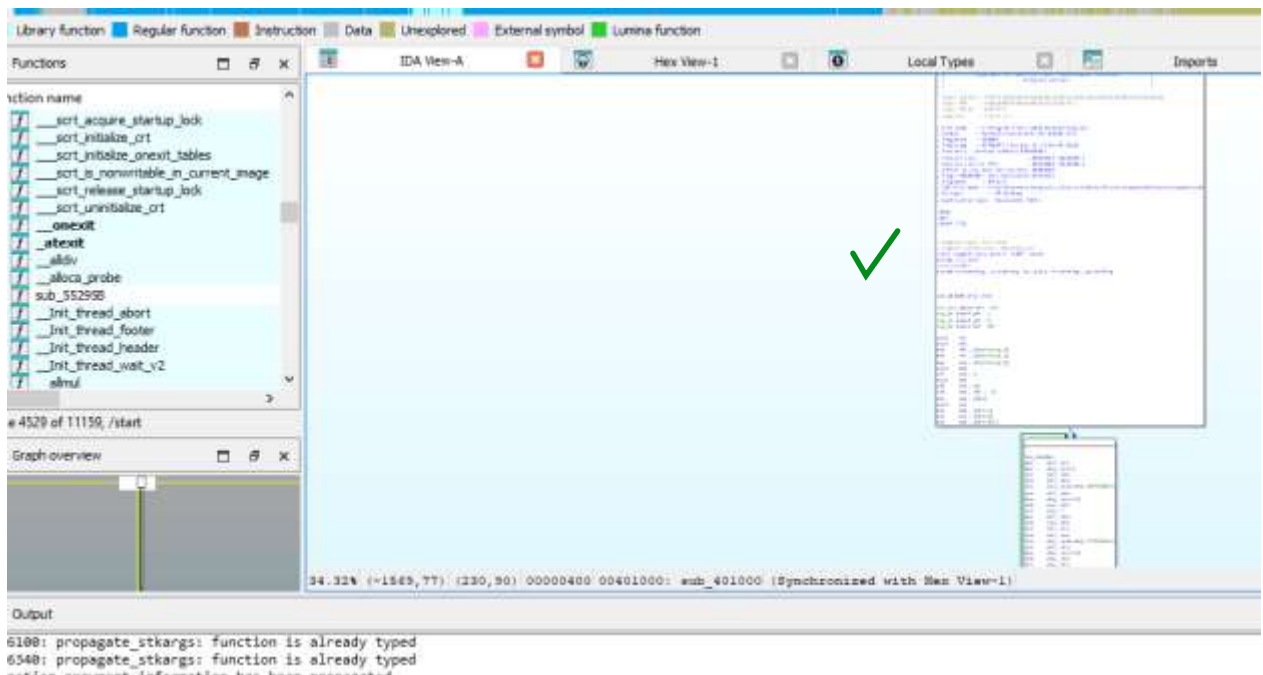
Copiamos la ruta y la ponemos en el ADI FREE



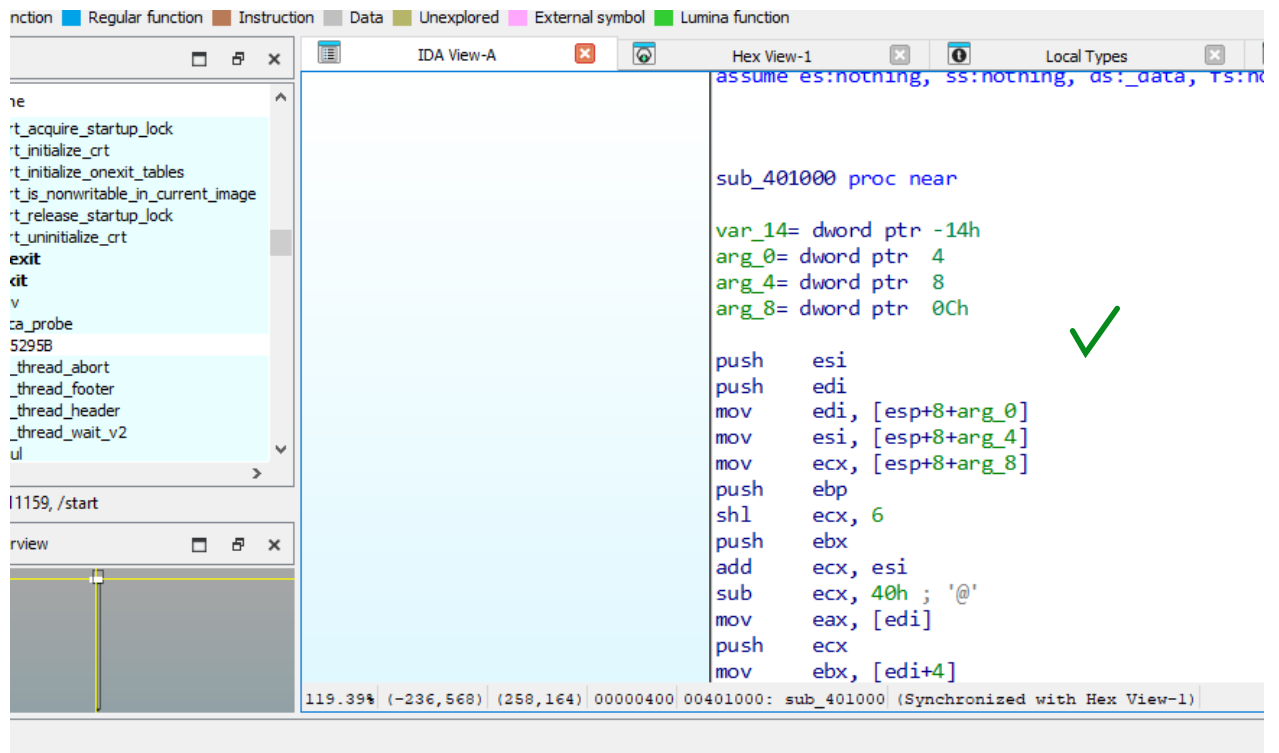
Ahora ponemos ok y procedemos a desensamblar el servidor



Como se puede ver aquí se tiene como una estructura de tablas



Se puede ver código Assembler



El procedimiento es bueno pero falta explicacion de lo que se esta haciendo en sus capturas y del codigo assembler