|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UNIVERSIDAD AUTÓNOMA “TOMAS FRÍAS” CARRERA DE INGENIERÍA DE SISTEMAS** | | | |  |
| **Materia:** | Arquitectura de computadoras (SIS-522) | | |
| **Nombre:** | Alan Cristopher Mamani Zelaya | | |
| **Docente: Auxiliar:** | Ing. Gustavo A. Puita Choque | | | N° Práctica |
| Univ. Aldrin Roger Perez Miranda | | | 9 |
| **20/10/2024** | **Fecha publicación** | | |
| **03/11/2024** | **Fecha de entrega** | | |
| **Grupo:** | **1** | **Sede** | **Potosí** | |

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

**utiliza?**

la pila es una estructura de datos fundamental en ensamblador que se utiliza para almacenar temporalmente valores y direcciones de memoria, implementar subrutinas, manejar interrupciones y más.

**2. Describe un escenario práctico donde el uso de ensamblador sería**

**más ventajoso que el uso de un lenguaje de alto nivel.**

**Ejemplo**

Supongamos que necesitas escribir una rutina de control para el marcapasos que monitore continuamente la señal eléctrica del corazón y responda en tiempo real. El ensamblador te permitirá:

* Configurar registros del microcontrolador para la lectura de señales analógicas de los electrodos.
* Implementar rutinas de interrupción precisas para gestionar eventos críticos del ritmo cardíaco.
* Optimizar el código para garantizar que las respuestas del marcapasos ocurran sin retraso, asegurando la seguridad del paciente.

**3. Explique cada línea del siguiente código del lenguaje ensamblador y**

**diga que es lo que se está haciendo**

lo que se está haciendo:

1. Se inicializan los registros AX y BX con los valores 5 y 10 respectivamente.
2. Se suman los valores de AX y BX, y el resultado (15) se almacena en AX.
3. Finalmente, el valor de AX (15) se copia al registro CX.

**4. Explique detalladamente cómo funcionan los compiladores**

Proceso

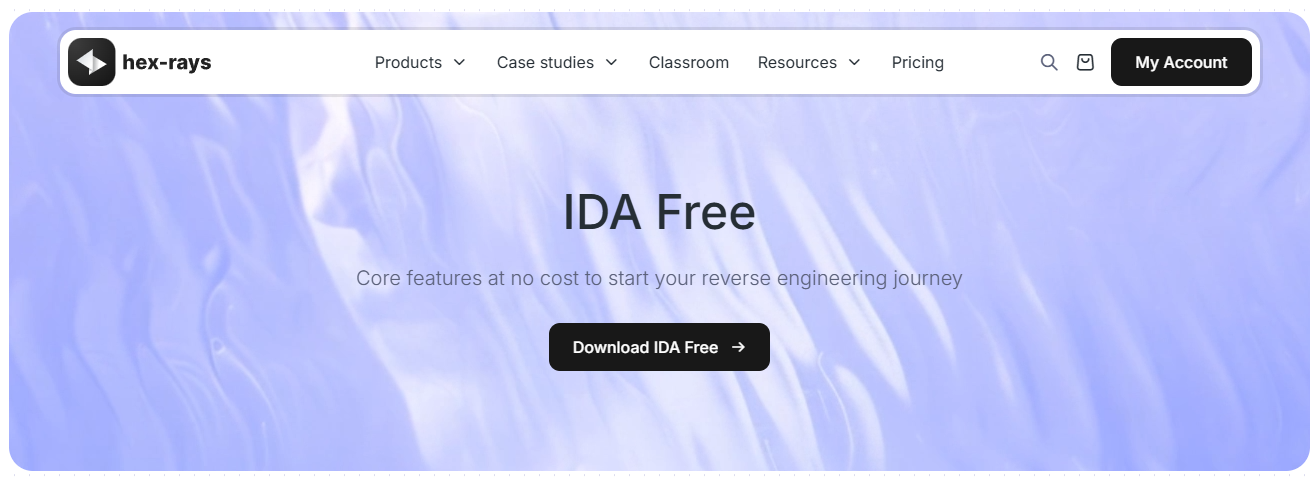
1. Análisis Léxico: Tokenización del código fuente.
2. Análisis Sintáctico: Construcción del árbol sintáctico.
3. Análisis Semántico: Verificación del significado y tipo.
4. Optimización: Mejoras en el código intermedio.
5. Generación de Código Intermedio: Creación de una representación intermedia.
6. Generación de Código Máquina: Transformación en código ejecutable.
7. Enlazado: Combina el código en un ejecutable final.

Este flujo garantiza que el código fuente de alto nivel sea traducido de manera precisa y eficiente a un formato que la máquina pueda entender y ejecutar.

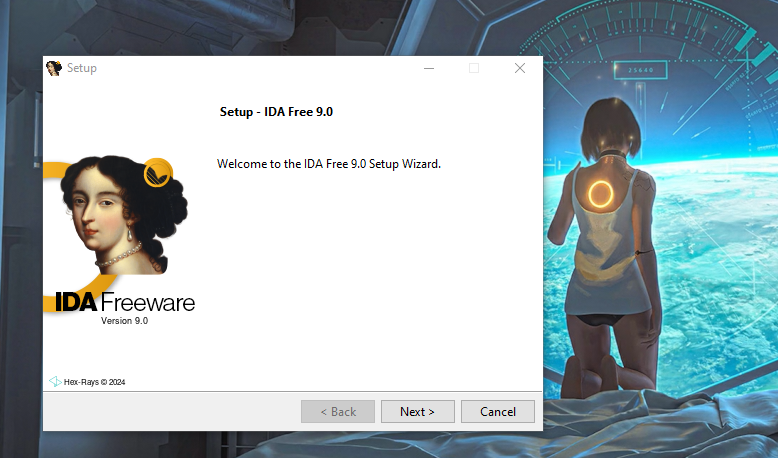
**5. Realizar sus propias capturas de pantalla del siguiente**

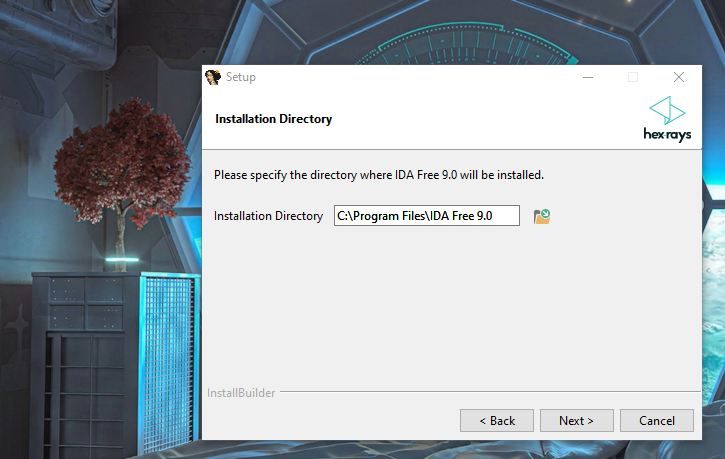
**procedimiento:**

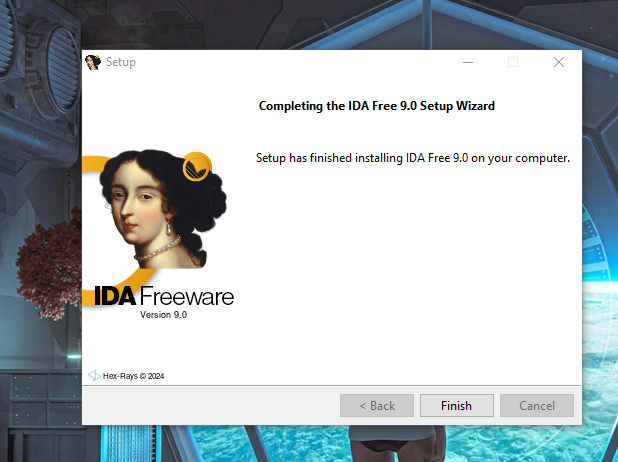
Paso 1:



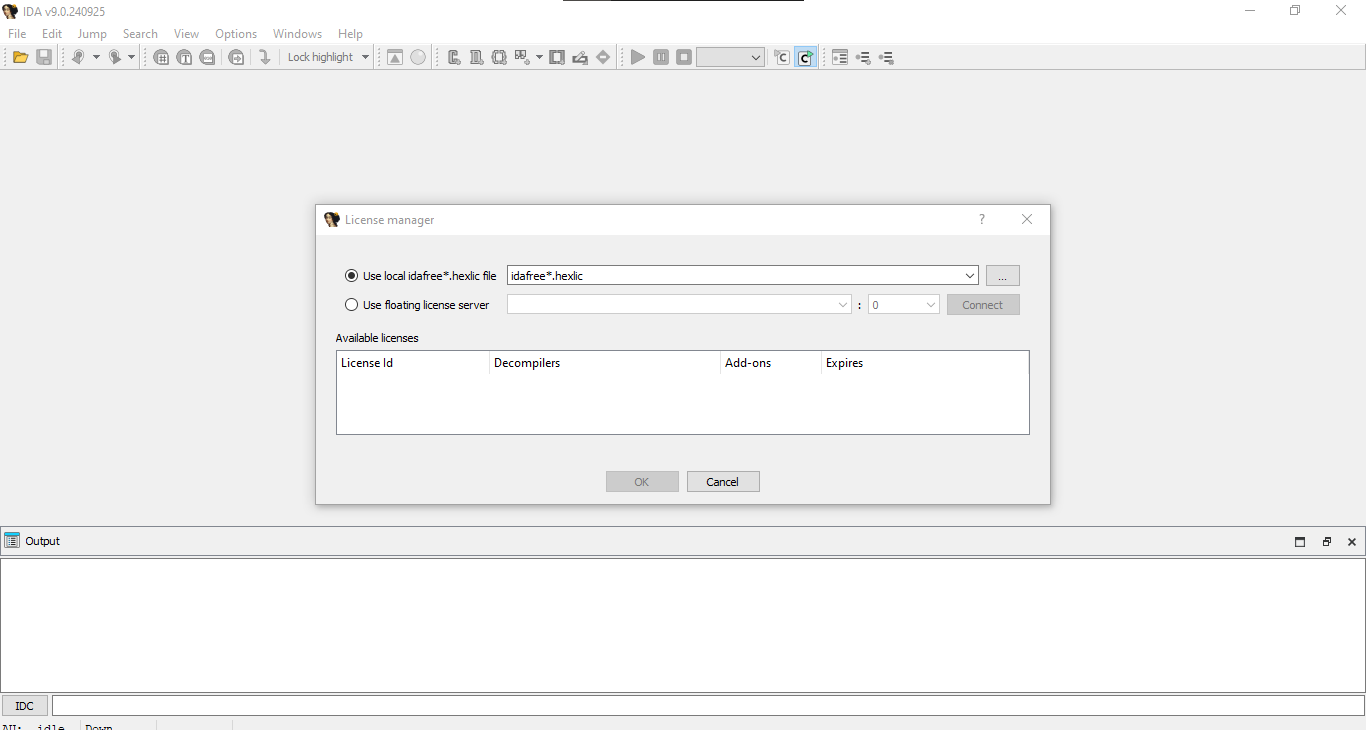
Paso 2:

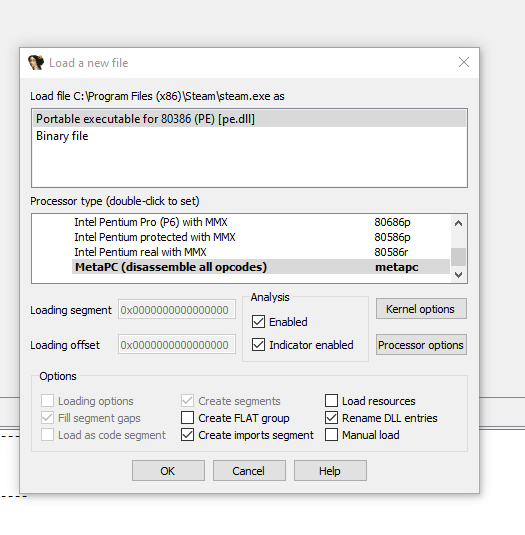






Paso 3:





Paso 4:

