# Práctica 4: Pregunta y Bonus

**Grupo:** GPT-Masters
**Profesor:** José Geralbert Rubiano

*Escuela de Ingeniería de Sistemas e Informática*
*Arquitectura de computadores - A2*
*Universidad Industrial de Santander*

3 de Octubre de 2024

## 1. Teniendo en cuenta las características del ensamblador, ¿Cuál es la principal limitante que observan? Justifique su respuesta.

Para empezar, debemos definir las características del ensamblador, las cuales son:

- Lenguaje de bajo nivel
- Depende de la arquitectura.
- Optimización de recursos.
- Difícil de mantener.
- Rapidez en la ejecución.

Al ser un lenguaje de bajo nivel está muy cercano al lenguaje máquina, por lo que las instrucciones las entiende directamente el procesador, a diferencia de lenguajes de alto nivel como Java.

Partiendo de que las instrucciones las entiende directamente el procesador, sus principales características surgen a partir de este hecho (debido a que el programador depende de la arquitectura del procesador y su conocimiento del mismo). En este punto se pueden visualizar sus ventajas y desventajas.

Como ventajas esta el hecho de que se puede escribir un código optimizado y eficiente, aprovechando la comunicación directa con el procesador, dado que se puede ajustar el código de manera que aproveche al máximo los recursos de la CPU. Este nivel de optimización es propio de los lenguajes de bajo nivel, ya que no requieren capas intermedias ni instrucciones adicionales para ejecutarse, lo que conlleva a que el tamaño del código se minimice. Debido a esto, se da la rapidez en la ejecución de los programas en ensamblador.

En el lado opuesto, se encuentran como desventajas que el código es menos legible debido al bajo nivel de abstracción que se maneja en ensamblador, además de que se depende mucho del conocimiento del programador al momento de hacer procesos básicos de programación: en lenguajes como Java se especifica el tipo de dato en el momento de la declaración, por lo que se obtendría un error al hacer algo como una multiplicación entre un dato tipo int y un string; por otro lado, en ensamblador podríamos hacer algo como una suma entre un número y un booleano, ya que el lenguaje se encarga de ejecutar instrucciones sin tener contexto de los datos, por lo que ensamblador permite ejecutar cosas sin sentido [2].

También se debe tener en cuenta que el set de instrucciones de cada programa depende de la arquitectura del procesador. Esto hace que los programas no sean portables entre distintos tipos de arquitecturas de hardware. Por ejemplo, un programa desarrollado para una arquitectura **Intelx86** no puede ser usado directamente en una arquitectura **ARM** sin realizar modificaciones profundas.

Después de haber analizado las ventajas y desventajas del lenguaje ensamblador, podemos definir como principal limitante la dependencia del hardware y su falta de portabilidad. En un mundo hiperglobalizado, con un mercado tan amplio, la oferta de componentes de hardware es muy alta, por lo que realizar un programa que funcione únicamente en un procesador especifico, entre la amplia gama que existe de estos, es una opción poco atractiva en un mercado donde se necesitan soluciones flexibles y escalables, por lo que consideramos que esa es la principal limitante del lenguaje ensamblador.[1]

# Bonus: ¿Por qué es tan importante el ensamblador?

Tomando como base la materia que cursamos en el momento: **arquitectura de computadores**, podemos explicar la importancia del lenguaje ensamblador desde el hecho de que

es el punto directo entre software y hardware; es decir, cómo mediante el software podemos darle instrucciones, casi directamente, a un componente del hardware, como lo es el procesador. Podemos verlo como la acción en la que el programador le está diciendo qué hacer a la máquina paso a paso, cosa que en los lenguajes de alto nivel no se hace directamente, ya que hay varias capas intermedias e instrucciones que interactúan en la conexión entre software y hardware.

Centrándonos más en el funcionamiento de ensamblador, podemos atribuirle gran importancia debido a que le ofrece al programador un control total de sus recursos, algo que no es común debido al uso de sistemas operativos, interfaces y demás aplicaciones que consumen recursos de maneras diferentes de las que, la mayoría de veces, el usuario final no se entera. Además, trabajar con este lenguaje le ofrece al programador la capacidad de comprender, desde la capa más profunda, cómo funciona un procesador y cómo se ejecutan los programas.

En conclusión, la importancia de ensamblador radica en que ofrece un **control detallado** de los recursos y un entendimiento profundo del sistema.[3]

# Referencias


[1] Castellanos Inma (2007). *Plataforma Acens para agencias interactivas*. https://www.acens.com/comunicacion/file_download/146/interactiva_plataforma_acens_para_agencias_interactivas.pdf

[2] Charles W. Kann III () *Implementing a One Address CPU in Logisim*. https://espanol.libretexts.org/Ingenieria/Implementacin_de_una_CPU_de_una_direccin_en_Logisim_(Kann)/02%3A_Lenguaje_de_la_Asamblea/2.02%3A_Advertencias_del_lenguaje_ensamblador

[3] Ed Jorgensen, Universidad of Navada . *Why Learn Assembly Language*. https://eng.libretexts.org/Bookshelves/Computer_Science/Programming_Languages/x86-64_Assembly_Language_Programming_with_Ubuntu_(Jorgensen)/01%3A_Introduction/1.03%3A_Why_Learn_Assembly_Language