* Introduce yourself to your peers and describe any experience you have had with reverse engineering or your interest in the subject. How did you come to enroll in this course?
* Discuss some of the rationale that the video provides for why you should learn assembly language.
* Then, add to those claims regarding why you also believe that it is important to learn assembly language and specifically for software reverse engineering.
* Learning assembly language in 2020 may seem outdated, but it helps improve computer understanding and decision-making for developers.

1. [[00:05]](https://www.youtube.com/watch?v=iYRl50gtprA&t=5) Learning assembly language in 2020 is a waste of time.  
   1. Many students complain about having to learn assembly language.
   2. Although not commonly used professionally, some computing programs still require knowledge of assembly language.
2. [[00:56]](https://www.youtube.com/watch?v=iYRl50gtprA&t=56) Learning assembly language has several benefits.  
   1. Understanding how computers work under the hood
   2. Improving awareness and making better decisions as software developers
   3. Gaining insights into performance differences and optimizing code
   4. Occasional practical use in specific situations
   5. Assembly is a human-readable machine language used by compilers
3. [[01:44]](https://www.youtube.com/watch?v=iYRl50gtprA&t=104) Compilation process in C  
   1. C code is compiled into assembly instructions
   2. The assembler converts assembly code into binary machine instructions
4. [[02:34]](https://www.youtube.com/watch?v=iYRl50gtprA&t=154) Compile code to assembly for better understanding and debugging
   1. Software vulnerabilities are common, so understanding assembly can help debug and fix issues
   2. Working with embedded systems often requires debugging at the machine code level
   3. Optimizing code for performance may require analyzing and fine-tuning assembly instructions
5. [[03:23]](https://www.youtube.com/watch?v=iYRl50gtprA&t=203) Learn assembly to understand function calls and basic arithmetic.  
   1. Writing handcrafted assembly can handle specific cases efficiently.
   2. Learning assembly helps in understanding how function calls work in C.
   3. Knowing basic assembly instructions like arithmetic operations is recommended.
6. [[04:12]](https://www.youtube.com/watch?v=iYRl50gtprA&t=252) Assembly language consists of bitwise operations, comparison operations, memory access instructions, and control flow instructions.  
   1. Bitwise operations include and, or, not, xor.
   2. Comparison operations include greater than, less than, and equivalence testing.
   3. Memory access instructions involve loads and stores to retrieve information from memory.
   4. Control flow instructions include branches and jumps, as well as built-in instruction support for call stack operations, function calls, and system calls.
   5. Different processors have different sets of instructions for assembly language.
7. [[05:02]](https://www.youtube.com/watch?v=iYRl50gtprA&t=302) Understanding how assembly languages work and convert code into machine code is key to becoming a better programmer.  
   1. Becoming an expert in assembly may not be necessary for most jobs.
   2. Focus on learning the fundamentals of assembly and its relationship to high-level languages.
8. [[05:50]](https://www.youtube.com/watch?v=iYRl50gtprA&t=350) Learn C programming by converting simple C programs to assembly instructions.  
   1. You can use your C compiler to write simple C programs and convert them to assembly instructions.
   2. Refer to your processor's assembly language documentation to understand the translations.
   3. Make sure to turn off optimizations to understand the code more accurately.