Compilation Process

2 min

Before we dive too deep into Assembly language, let's take a step back and learn where Assembly language fits into the everyday process of

Preview: Docs Software engineering is a field of study that involves using engineering principles to design, build, test, and maintain software systems.

software engineering

.

When we push **Run** in our

Preview: Docs A code editor is a program designed for writing software and utilizing human-readable text to make code easier to parse and understand.

code editor

, the computer is performing a number of tasks in the background before the

Preview: Docs The physical components that comprise computer systems.

computer hardware

even gets to see its first bit of

Preview: Docs Loading link description

binary

code.

In general, there are four steps, known as the Compilation Process, that make up the journey high-level code goes on before reaching the hardware:

- **Preprocessing** is the first step of compilation and is used to prepare the user's code for

Preview: Docs Loading link description

machine code

by removing comments, expanding included macros, and performing any code maintenance prior to handing the file to the compiler.

- **Compiling** is the process of taking the expanded file from the preprocessor and translating the program into an optimized Assembly language.
- **Assembling** is the process of taking an assembly language program and using an assembler to generate machine code.
- **Linking** is the process of filling in function calls, including additional objects, libraries, and source code from other locations into the main source code.

While the compilation process is tailored to each language and architecture, the overall procedure is fairly standard. It is in the compiling and assembling stages where Assembly is generated and used to create machine code.

Instructions

1. Checkpoint 1 Passed

1.

Click **Run** to see the translation of the Python function in the code editor into the MIPS Assembly language. This is step two of the compilation process, compiling.

script.py

```
from translate import translate


# Software Developer's Code


def square(num_to_square):
    return num_to_square * num_to_square






translate()
```

translate.py

```
def translate():
    print("MIPS Assembly Language")
    print()
    print("square:")
    print("   addiu $sp,$sp,-8")
    print("   sw $fp,4($sp)")
    print("   move $fp,$sp")
    print("   sw $4,8($fp)")
    print("   lw $3,8($fp)")
    print("   lw $2,8($fp)")
    print("   nop")
    print("   mult $3,$2")
    print("   mflo $2")
    print("   move $sp,$fp")
    print("   lw $fp,4($sp)")
    print("   addiu $sp,$sp,8")
    print("   j $31")
    print("   nop")
```