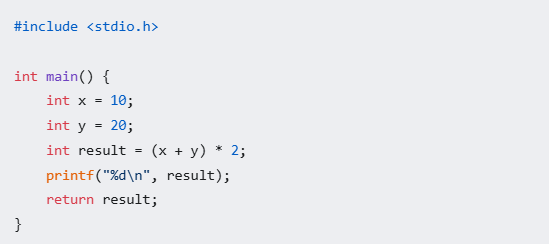
**Q2: Give the input and output of the miniC compiler?**

Ans:

**1. Input: Mini-C Source Code File (input.c)**

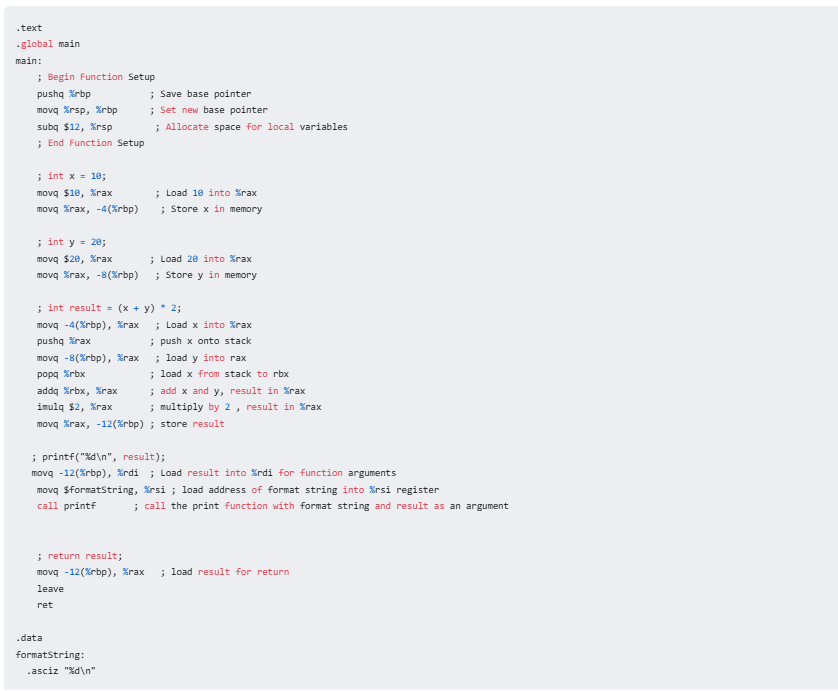
* **Input Type:**
  + **File Type:** Plain text file.
  + **File Extension:** .c
  + **Explanation:** The input is a text file specifically designated for Mini-C source code. The .c extension is a convention that helps both the user and the compiler recognize it as a file containing code written in the Mini-C language.



This source code represents a program that declares integer variables, performs a calculation, prints the result, and then returns the result, where the printf function has been used instead of the print statement from before, as this more accurately shows the required output, and a #include <stdio.h> statement has been added, to allow for the function printf to be used.

**Output: x86-64 Assembly Code File (output.s)**

* **Output Type:**
  + **File Type:** Plain text file.
  + **File Extension:** .s
  + **Explanation:** The output is a text file that contains x86-64 assembly language instructions. The .s extension is a standard convention that signifies assembly language code, that is meant for a specific computer architecture, such as x86-64.



The output file represents the program in the form of x86-64 assembly instructions. These instructions perform actions that are equivalent to the high-level source code, such as setting up the stack frame, memory access, performing the calculation, calling the print function, and returning a result.

The assembly code directly translates Mini-C program into machine-understandable instructions.

1. **Setup:** It sets up the function's stack space to store variables, using push, movq, subq instructions and stack offsets from %rbp.
2. **Calculation:** It performs the arithmetic operations (+, \*) using registers, the stack, and instructions like movq, addq, and imulq.
3. **Print:** It prepares arguments, and then makes a function call for printf using movq and call instructions.
4. **Return:** It loads the final result into register %rax, cleans up the stack, and uses the leave and ret instructions to exit the function.