How are Arguments Pushed in C?

What is the order of parameters?

For example, given the call

```
my_func (A, B, C);
```

Should a compiler place
- **A** on top of the stack?
- Or **C** on top of the stack?
- Or does the choice not matter?

Arguments Must Be Passed in the Right Order

First answer: Of course it matters!

How could one compiler generate the call,
and a second compiler generate the function,
if the order were not fixed?

Again, How are Arguments Pushed in C?

```
my_func (A, B, C);
```

So, do arguments get pushed
- **left to right,** or
- **right to left?**

You should be able to answer.

Remember that **C** functions
- can accept variable numbers of parameters.
- and must be able to figure out how many arguments have been passed.

Compilers Can Optimize within a Function

Stack frame use **inside a function**
- is not an interface issue,
- so **compilers can optimize**.

For example, compilers can
- place variables in registers,
- avoid saving and restoring R7 (for example, if no subroutines are called), or
- avoid creating a stack frame at all!

