Template Parameter: Variadic Templates

In this lesson, we will learn about one of the powerful new tools of C++11: variadic templates.

WE'LL COVER THE FOLLOWING

- Parameter pack

A **variadic template** is a template with an arbitrary number of parameters.

```
template <typename ... Args>
void variadicTemplate(Args ... args){ . . . . }
```

Parameter pack

- By using the ellipse (...) `Args-` or `args` becomes a parameter pack
- `Args` is a template parameter pack, and `args` is a function parameter pack
- Parameter packs can only be packed or unpacked
- If the ellipse is left from `Args`, the parameter pack will be packed. If the ellipse is right from Arge, the parameter pack will be unpacked.

> The compiler can automatically deduce the template arguments.

Variadic Templates are often used in the Standard Template Library:

- `sizeof-Operator`, `std::tuple`, `std::thread`

The usage of parameter packs obeys a typical pattern.

- Perform an operation on the first element of the parameter pack and recursively invoke the operation on the remaining elements.
- The recursion ends after a finite number of steps.

- The boundary condition is typically a fully specialized template.

```
template<>
struct Mult<>{ ... }
template<int i, int ... tail >
struct Mult<i, tail ...>{ ...
```

In the next lesson, we will examine examples of variadic templates.