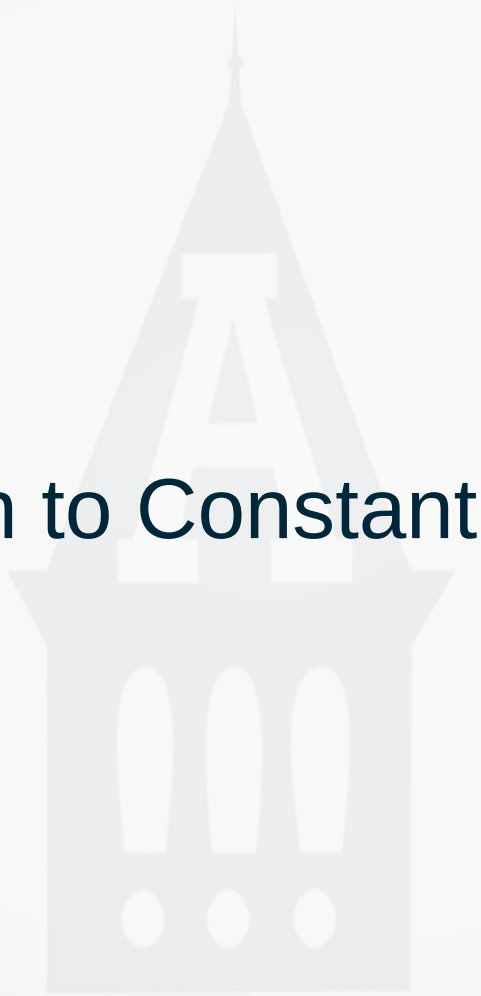
CS 3460



Introduction to Constant Expressions

Constant Expressions

- We already understand an expression
  - Translated into code, executed at run-time
  - Makes sense, get data at run-time, therefore, execute at run-time

Constant Expressions

- We already understand an expression
  - Translated into code, executed at run-time
  - Makes sense, get data at run-time, therefore, execute at run-time
- There are times, interestingly, when the data is known at compile-time
  - Therefore, compiler can evaluate at compile-time, placing the result directly in code
    - ***No*** run-time cost
  - C++ calls these *constant expressions*

Constant Expressions

- Keyword `const` means the value is known at time code is generated
- Keyword `constexpr` means the value is known during compile-time

Simple Example

- Let’s take a look at a simple example

```
constexpr auto sum(int a, int b)
{
    return a + b;
}

int main()
{
    constexpr auto total = sum(2, 6);
    std::cout << total << std::endl;

    return 0;
}
```

Simple Example

- Let’s take a look at a simple example

```
constexpr auto sum(int a, int b)
{
    return a + b;
}

int main()
{
    constexpr auto total = sum(2, 6);
    std::cout << total << std::endl;

    return 0;
}
```

- Note `constexpr` on both the function and expression

Code Demonstration – Generated ASM (use godbolt.org)

Another Example

- Good old recursive Fibonacci

```
constexpr auto fibonacci(unsigned int n)
{
    if (n == 0 || n == 1)
        return 1;

    return fibonacci(n - 1) + fibonacci(n - 2);
}


constexpr auto fibN = fibonacci(11);
std::cout << fibN << std::endl;
```

- Inspection of compiler output shows 144; the correct value
- Notice it evaluates a recursive function!

Another Example – Continued

- Can still use a constexpr for run-time evaluation

```
std::cout << "Enter a Fibonacci number to compute: ";
std::uint16_t input;
std::cin >> input;
std::cout << "The value is: " << fibonacci(input) << std::endl;
```

`const` or `constexpr` for constants?

- Reference discussion
  - https://stackoverflow.com/questions/14116003/difference-between-constexpr-and-const
- Consider we have two golden ratios

```
auto const GOLDEN_RATIO = 1.6180339887;
auto constexpr GOLDEN_RATIO = 1.6180339887;
```

- Both ensure the value cannot be modified
- The `constexpr` version is known at compile-time
  - Can be used during compile-time evaluation
- `const` is usually fine, but sometimes `constexpr` is necessary

Code Demo – Improving Switch