- Examples

In this lesson, we'll take a look at the examples of implementing constexpr functions.

WE'LL COVER THE FOLLOWING

- Example 1
  - Explanation
- Example 2
  - Explanation

Example 1

```
//C++ 11
// constExpression.cpp
#include <iostream>

constexpr int square(int x) { return x * x; }
constexpr int squareToSquare(int x){ return square(square(x));}

int main() {

  std::cout << std::endl;

  int a = 10;
  static_assert(square(a) == 100, "you calculated it wrong");
  static_assert(squareToSquare(10) == 10000 , "you calculated it wrong");

  std::cout<< "square(10)= " << square(10) << std::endl;
  std::cout<< "squareToSquare(10)= " << squareToSquare(10) << std::endl;

  constexpr int constExpr= square(10);

  int arrayClassic[100];
  int arrayNewWithConstExpression[constExpr];
  int arrayNewWithConstExpressioFunction[square(10)];

  std::cout << std::endl;

}
```

Explanation

- In the example above, we implemented two `constexpr` functions for C++11: `constexpr int square(int x)` and `constexpr int squareToSquare(int x)`. Both functions follow the conventions for `constexpr` functions definition in C++11.
- The assertation in lines 13 and 14 fail since `a` is not a literal type. Making `a` `constexpr` variable enables the code compilation to pass those assertations.
- In line 19, we initialized a `constexpr` variable `constExpr` using the `square` function.
- In lines 21-23, we have initialized three arrays in three ways:
    1. by using a constant 100.
    2. by using a `constexpr` variable `constExpr`.
    3. by calling the function `square(10)`. Notice that the input argument for this function call is a constant.

Example 2

```
//constExpressionCpp14.cpp
#include <iostream>

constexpr int gcd(int a, int b){
  while (b != 0){
    auto t= b;
    b= a % b;
    a= t;
  }
  return a;
}

int main(){

  constexpr auto res= gcd(100, 10);
  std::cout << "gcd(100, 10) " << res << std::endl;

}
```

Explanation

- The difference between ordinary functions and `constexpr` functions in C++14 is minimal. It is quite easy to implement the `gcd` algorithm in C++14 as a `constexpr` function.
- We have defined `res` as a `constexpr` variable, and its type is automatically determined by `auto`.
- Note that we cannot use a non-constant value as input arguments for the `gcd` function.

Test your knowledge of this topic with an exercise in the next lesson.