```
       constexpr int f1() { return 0; } // automatically inline
inline constexpr int f1();              // redeclares the same f1() above
```

As with all **inline** functions, it is a **one-definition rule (ODR)** violation if definitions in different **translation units** within a program are not token for token the same. If definitions do differ across translation units, the program is IFNDR:

```
// file1.h:
          inline int f2() { return 0; }
       constexpr int f3() { return 0; }

// file2.h:
          inline int f2() { return 1; }  // Error, no diagnostic required
       constexpr int f3() { return 1; }  // Error, no diagnostic required
```

When a function is declared **constexpr**, *every* declaration of that function, including its definition, must also be explicitly declared **constexpr**, or else the program is ill formed:

```
constexpr int f4();
constexpr int f4() { return 0; }  // OK, constexpr matching exactly

constexpr int f5();
          int f5() { return 0; }  // Error, constexpr missing

          int f6();
constexpr int f6() { return 0; }  // Error, constexpr added
```

An **explicit specialization** of a **function template** declaration may, however, differ with respect to its **constexpr** specifier. For example, a general function template, e.g., `ft1` in the code snippet below, might be declared **constexpr**, whereas one of its explicit specializations, e.g., `ft1<int>`, might not be:

```
template <typename T>    // general function template declaration/definition
constexpr bool ft1(T)    // general template is declared constexpr
{
    return true;
}

template <>              // explicit specialization declaration/definition
bool ft1<int>(int)       // The explicit specialization is not constexpr.
{
    return true;
}

static_assert(ft1('a'), "");  // OK, general function template is constexpr.
static_assert(ft1(123), "");  // Error, int specialization is not constexpr.
```

Similarly, the roles can be reversed where only an explicit specialization, e.g., `ft2<int>` in the next example, is **constexpr**: