【C++】Day three

Class	C++
Date	@November 10, 2021
Material	
Series Number	
Summary	

【Ch2】Primitive Built-in Types

C++ defines a set of primitive types that include the arithmetic types and a special type named void.

- The arithmetic types represent characters, integers, boolean values and floating-point numbers.
- The void type has no associated values and can be used in only a few circumstances, most commonly as the return type for functions that do not return a value.

2.1.1 Arithmetic Types

The arithmetic types are divided into two categories: integral types(which include character and boolean types) and floating-point types.

Notice:

1. Do not use plain char or bool in arithmetic expressions. Use them only to hold characters or truth values.

   Computations using char are especially problematic because char is signed on some machines and unsigned on others. If a tiny integer is needed, explicitly specify either signed char or unsigned char.

2. Use double for folating-point computations; float usually does not have enough precision, and the cost of double-precision calculations versus single-precision is

negligible.

2.1.2 Type Conversions

See the following code:

```
bool a = 42;
int b = a;
```

When we assign a bool to one of the other arithmetic types, the resulting value is 1 if the bool is true and 0 if the bool is false.

By the same token, when we use a bool in an arithmetic expression, its value always converts to either 0 or 1. As a result, using a bool in an arithmetic expression is almost surely incorrect.

2.1.3 Literals

A value, such as 42, is known as a literal because its value is self-evident.

Integer and Floating-Point Literals

We can write an integer literal using decimal, octal, or hexadecimal notation.

- Integer literals that begin with 0 are interpreted as octal.
- Integer literals that begin with 0x or 0X are interpreted as hexadecimal

Different ways of defining 20.

```
int a = 20;
int b = 024;
int c = 0x14;
```

String Literals

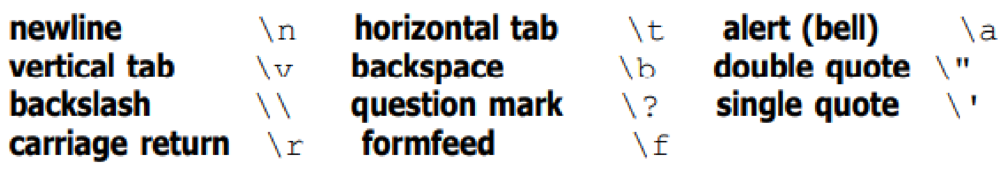
Two string literals that appear adjacent to one another and that are separated only by spaces, tabs, or newlines are concatenated into a single literal.

See the following code for an example:

```
std::cout << "A really really long "
             "String" << std::endl;
```

Special Characters

Some characters are not printable as they have other meanings in the language. To print them, put a \(backslash) in front of these characters.



newline	\n	horizontal tab	\t	alert (bell)	\a
vertical tab	\v	backspace	\b	double quote	\"
backslash	\\	question mark	\?	single quote	\'
carriage return	\r	formfeed	\f		