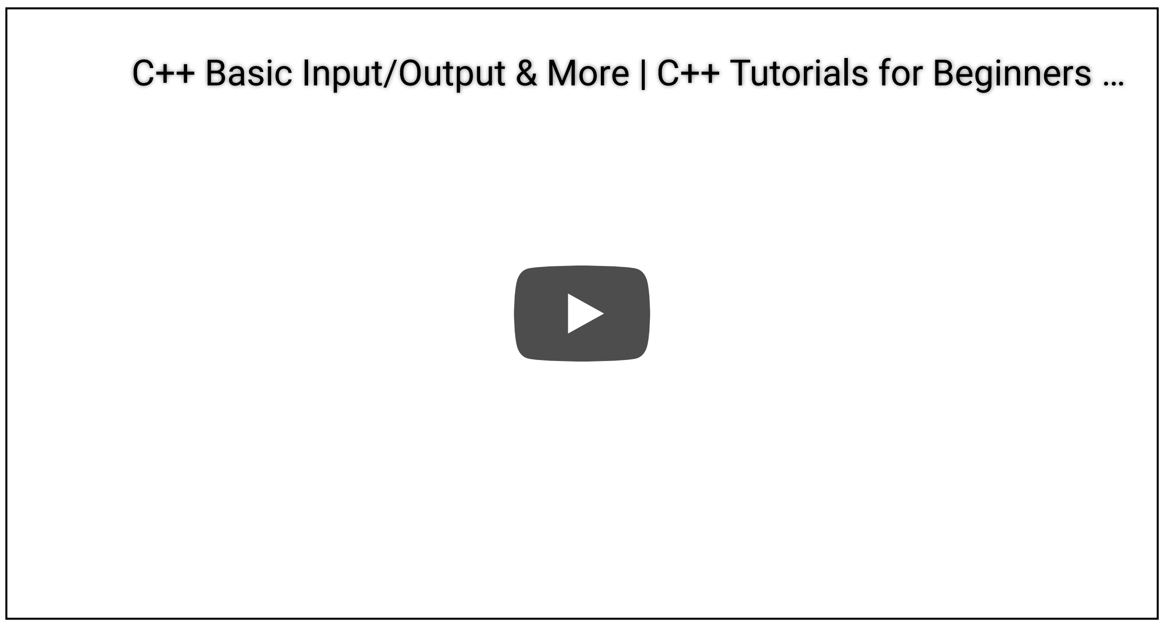

C++ Basic Input/Output & More | C++ Tutorials for Beginners ...

Overview **Q&A** **Files** **Announcements**

C++ Basic Input/Output & More | C++ Tutorials for Beginners #5

In this tutorial, we will visualize basic input and output in the C++ language. In our last lesson, we discussed the variable's scope and data types. In this C++ tutorial, we are going to cover basic input and output:

Basic Input and Output in C++

C++ language comes with different libraries, which helps us in performing input/output operations. In C++ sequence of bytes corresponding to input and output are commonly known as streams. There are two types of streams.

Input stream

In the input stream, the direction of the flow of bytes occurs from the input device (for ex keyboard) to the main memory.

Output stream

In output stream, the direction of flow of bytes occurs from main memory to the output device (for ex-display)

Practical Explanation of Input/Output

We will see the actual code for input/output, and it's working. Consider the code below:

```
# include<iostream>
using namespace std;

int main()
{
    int num1, num2;
    cout<<"Enter the value of num1:\n"; /* '<<' is called Insertion operator */
    cin>>num1; /* '>>' is called Extraction operator */

    cout<<"Enter the value of num2:\n"; /* '<<' is called Insertion operator */
    cin>>num2; /* '>>' is called Extraction operator */

    cout<<"The sum is "<< num1+num2;

    return 0;
}
```

Figure 1: Basic input/output program

Course Content

Hide Player

In this piece of code, we have declared two integer variables "**num1**" and "**num2**". Firstly we used "**cout**" to print "**Enter the value of num1**:" as it is on the screen, and then we used "**cin**" to take the input in "**num1**" at run time from the user.

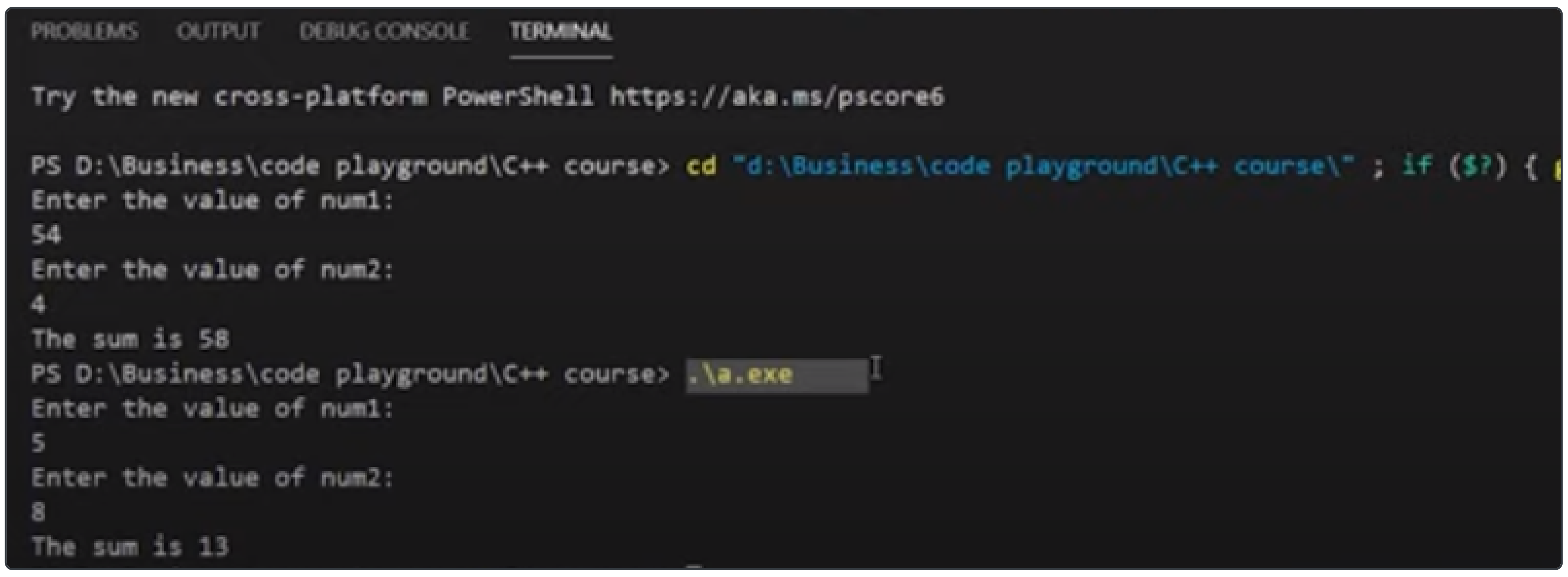
7. C++ Reference Variables & Typecasting | C++ Tutorials for Beginners #7
Free YouTube Video

Secondly, we used "**cout**" to print "**Enter the value of num2**:" as it is on the screen, and then we used "**cin**" to take the input in "**num2**" at run time from the user.

Never take your phone out.
On-board Google maps navigation.
Ather 450X. Quick. Electric. Intelligent.
Book Now
Google and the registered trademark
Vehicle images shown are for representation

In the end, we used "**cout**" to print "**The sum is**" as it is on the screen and also gave the literal "**num1+num2**" which will add the values of both variables and print it on the screen.

The output of the following program is shown in figure 2.



```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Business\code playground\C++ course> cd "d:\Business\code playground\C++ course\" ; if ($?) {
Enter the value of num1:
54
Enter the value of num2:
4
The sum is 58
PS D:\Business\code playground\C++ course> .\a.exe
Enter the value of num1:
5
Enter the value of num2:
8
The sum is 13
```

Figure 2: Output of the Program

We have executed our program two times, which can be seen in figure 2. In our 1st execution, we had input the value "**54**" for the variable "**num1**" and value "**4**" for the variable "**num2**". This gives us the sum of both numbers as "**58**".

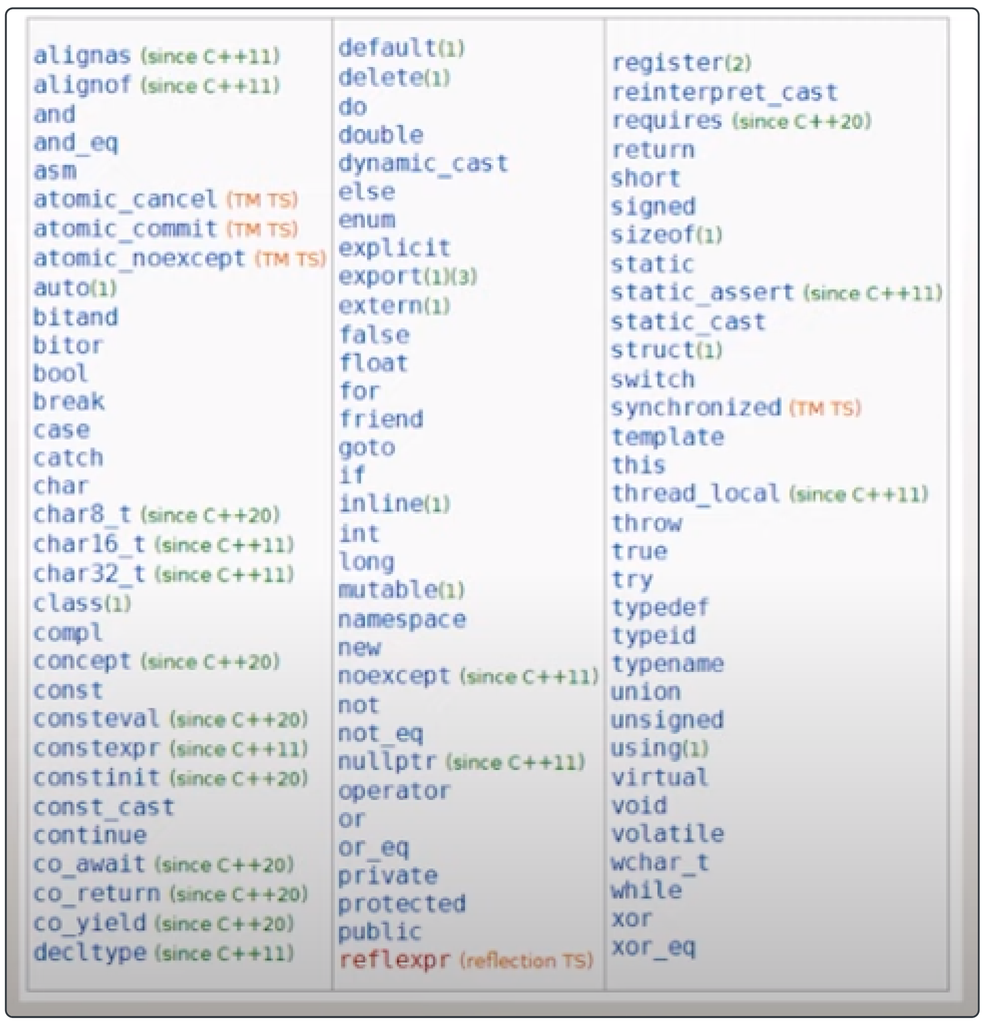
In our 2nd execution, we had input the value "**5**" for the variable "**num1**" and value "**8**" for the variable "**num2**". This gives us the sum of both numbers as "**13**".

Important Points

1. The sign "**<<**" is called insertion operator
2. The sign "**>>**" is called extraction operator
3. "**cout**" keyword is used to print
4. "**cin**" keyword is used to take input at run time.

Reserved keywords in C++

Reserved keywords are those keywords that are used by the language itself, which is why these keywords are not available for re-definition or overloading. In short, you cannot create variables with these names. A list of reserved keywords is shown in figure 3.



alignas (since C++11)
alignof (since C++11)
and
and_eq
asm
atomic_cancel (TM TS)
atomic_commit (TM TS)
atomic_noexcept (TM TS)
auto(1)
bitand
bitor
bool
break
case
catch
char
char8_t (since C++20)
char16_t (since C++11)
char32_t (since C++11)
class(1)
compl
concept (since C++20)
const
consteval (since C++20)
constexpr (since C++11)
constinit (since C++20)
const_cast
continue
co_await (since C++20)
co_return (since C++20)
co_yield (since C++20)
decltype (since C++11)

default(1)
delete(1)
do
double
dynamic_cast
else
enum
explicit
export(1)(3)
extern(1)
false
float
for
friend
goto
if
inline(1)
int
long
mutable(1)
namespace
new
noexcept (since C++11)
not
not_eq
nullptr (since C++11)
operator
or
or_eq
private
protected
public
reflexpr (reflection TS)

register(2)
reinterpret_cast
requires (since C++20)
return
short
signed
sizeof(1)
static
static_assert (since C++11)
static_cast
struct(1)
switch
synchronized (TM TS)
template
this
thread_local (since C++11)
throw
true
try
typedef
typeid
typename
union
unsigned
using(1)
virtual
void
volatile
wchar_t
while
xor
xor_eq

Figure 3: Reserved keywords in C++

Code as described/written in the video

```
# include<iostream>
using namespace std;

int main()
{
    int num1, num2;
    cout<<"Enter the value of num1:\n"; /* '<<' is called Insertion operator */
    cin>>num1; /* '>>' is called Extraction operator */

    cout<<"Enter the value of num2:\n"; /* '<<' is called Insertion operator */
    cin>>num2; /* '>>' is called Extraction operator */

    cout<<"The sum is "<< num1+num2;

    return 0;
}
```

← Previous

Next →

Back to top | Privacy | Terms