**void pointers.**

The void type of pointer is a special type of pointer. In C++, void represents the absence of type. Therefore, void pointers are pointers that point to a value that has no type (and thus also an undetermined length and undetermined dereferencing properties).

This gives void pointers a great flexibility, by being able to point to any data type, from an integer value or a float to a string of characters. In exchange, they have a great limitation: the data pointed to by them cannot be directly dereferenced (which is logical, since we have no type to dereference to), and for that reason, any address in a void pointer needs to be transformed into some other pointer type that points to a concrete data type before being dereferenced.

One of its possible uses may be to pass generic parameters to a function. For example:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 | // increaser  #include <iostream>  using namespace std;  void increase (void\* data, int psize)  {  if ( psize == sizeof(char) )  { char\* pchar; pchar=(char\*)data; ++(\*pchar); }  else if (psize == sizeof(int) )  { int\* pint; pint=(int\*)data; ++(\*pint); }  }  int main ()  {  char a = 'x';  int b = 1602;  increase (&a,sizeof(a));  increase (&b,sizeof(b));  cout << a << ", " << b << '\n';  return 0;  } | y, 1603 | [Edit & Run](https://www32.cplusplus.com/doc/tutorial/pointers/) |

*sizeof* is an operator integrated in the C++ language that returns the size in bytes of its argument. For non-dynamic data types, this value is a constant. Therefore, for example, *sizeof(char)* is 1, because char has always a size of one byte.