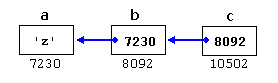
**Pointers to pointers.**

C++ allows the use of pointers that point to pointers, that these, in its turn, point to data (or even to other pointers). The syntax simply requires an asterisk (\*) for each level of indirection in the declaration of the pointer:

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 5 6 | char a;  char \* b;  char \*\* c;  a = 'z';  b = &a;  c = &b; |  |

This, assuming the randomly chosen memory locations for each variable of 7230, 8092, and 10502, could be represented as:



With the value of each variable represented inside its corresponding cell, and their respective addresses in memory represented by the value under them.

The new thing in this example is variable *c*, which is a pointer to a pointer, and can be used in three different levels of indirection, each one of them would correspond to a different value:

* c is of type char\*\* and a value of 8092
* \*c is of type char\* and a value of 7230
* \*\*c is of type char and a value of 'z'

Check below program:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17  18  19 | #include <iostream>  using namespace std;  int main () {  char a;  char \*b;  char \*\*c;  a = 'z';  b = &a;  c = &b;  cout << c << " == " << c << endl;  cout << \*c << " == " << b << endl;  cout << \*\*c << " == " << a << endl;  cout << \*b << " == " << a << endl;  cout << &c << " != " << &a << endl;  cout << c << " != " << &a << endl;  cout << c << " == " << &b << endl;  cout << b << " != " << &c << endl;  cout << b << " == " << &a << endl;  } | 00A1FD68 == 00A1FD68  z╝¤б == z╝¤б  z == z  z == z  00A1FD6C != z╝¤б  00A1FD68 != z╝¤б  00A1FD68 == 00A1FD68  z╝¤б != 00A1FD6C  z╝¤б == z╝¤б | [Edit & Run](https://www32.cplusplus.com/doc/tutorial/pointers/) |