• Explain what is meant by "local variable" in C++

* Variable that defied in a scope wrapped with braces, it’s created in the memory when the scope started and deleted with the scope end. If you didn’t initialize it, it will be initialized by garbage.

• In the following code, what is the initial value of d? double d;

* Garbage

• Explain what is meant by the following terms:

* Pass by value
  + The function parameter will be normal local variable/object that carry a copy from the passed value. Any change at the parameter will affect only the parameter and will end with the function end unless u return this parameter after update.
* Pass by address
  + Send the address to a pointer-parameter, any changes in the parameter will affect the essential variable/object that’s address was passed. The called function will create a pointer to receive this address at.
* Pass by reference
  + This behaves like pointer but there will not be a real pointer created in the memory. The caller function will pass a normal variable and the called function will have a reference to the passed value which is automatically dereferenced when use it.

• Add code to your programs to show the addresses of the variables involved. Explain your observations

#include <iostream>

using namespace std;

int passByValue(int y) {

cout << "Address of y is " << &y << endl;

cout << "value of y is " << y << endl;

y = 5;

cout << "value of y after modifications is " << y << endl;

return y;

}

int passByAddress(int\* y) {

cout << "Address of y is " << &y << endl;

cout << "value of y is " << \*y << endl;

\*y = 11;

cout << "value of y after modifications is " << \*y << endl;

return \*y;

}

int passByReference(int& y) {

cout << "Address of y is " << &y << endl;

cout << "value of y is " << y << endl;

y = 15;

cout << "value of y after modifications is " << y << endl;

return y;

}

int main() {

int x = 2;

cout << "Address of x is " << &x << endl;

cout << "value of x is " << x << endl;

cout << endl;

int z = 0;

cout << "Address of z is " << &z << endl;

cout << "value of z is " << z << endl;

cout << endl;

z = passByValue(x);

cout << "After calling passByValue(), z = " << z << ", x = " << x << endl;

cout << "Address of x is " << &x << endl;

cout << "Address of z is " << &z << endl;

cout << endl;

z = passByAddress(&x);

cout << "After calling passByAddress(), z = " << z << ", x = " << x << endl;

cout << "Address of x is " << &x << endl;

cout << "Address of z is " << &z << endl;

cout << endl;

z = passByReference(x);

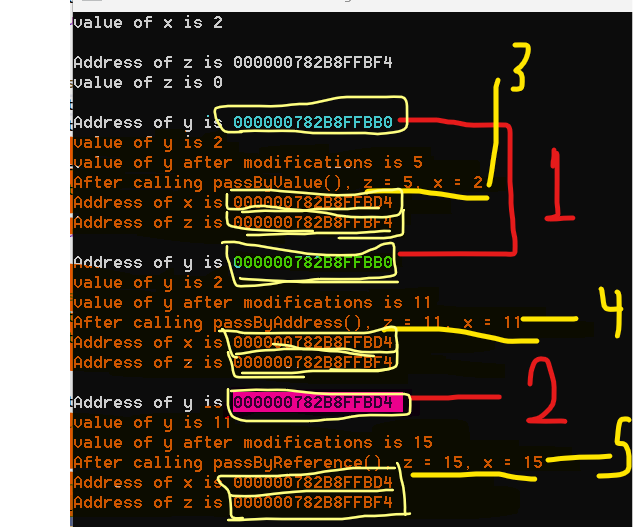
cout << "After calling passByReference(), z = " << z << ", x = " << x << endl;

cout << "Address of x is " << &x << endl;

cout << "Address of z is " << &z << endl;

cout << endl;

}



* 1 and shows the addresses difference. 1 shows that there’s a variable created but 2 shows that it’s the same address only names are different like x and y... etc.
* 3, 4,5 shows the value change.