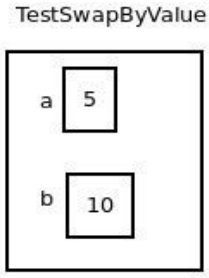
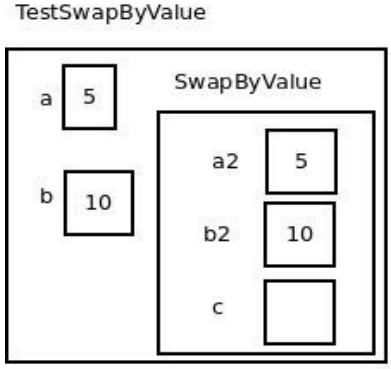
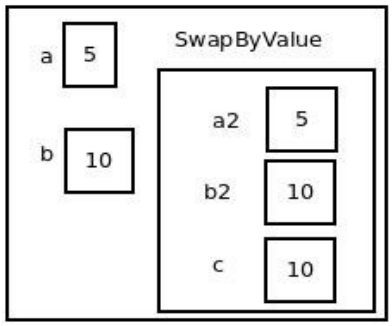
```
#include <string>
#include <iostream>

using namespace std;

//This function swaps the content of two variables.
//@post This takes in copies of some given arguments
//    and swaps those values within the scope of this function
//@param a2 This value is a copy of the original one that was passed.
//@param b2 This value is a copy of the original one that was passed.
void swap_by_value ( int a2 , int b2 )
{
     int tmp = a2;
     a2 = b2;
     b2 = tmp;
};

//This is meant to test the swap_by_value() function.
//@post This will print the result of the test for the swap_by_value()
function
//    to the console.
void testByValue()
{
     int a = 10;
     int b = 15;

     cout << "Before: a) " << a << " " << "b) " << b << endl;

     swap_by_value(a,b);

     cout << "After : a) " << a << " " << "b) " << b << endl;
};
```



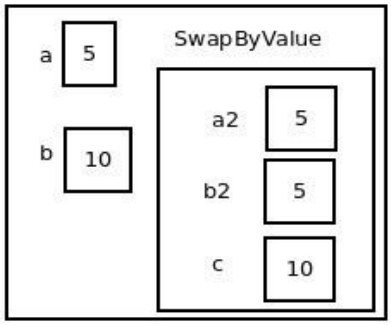
This is right before SwapByValue is called.



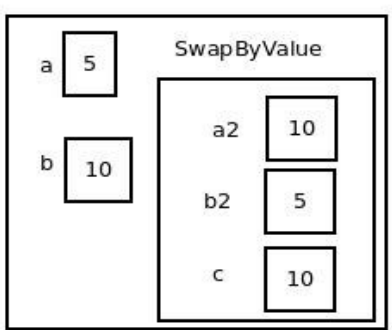
This is right after SwapByValue is called within the TestSwapByValue function.



This is right after variable b2 is assigned to c.



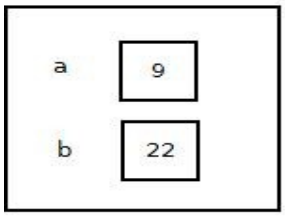
This is right after variable a2 is assigned to b2.



This is right after variable c is assigned to a2.

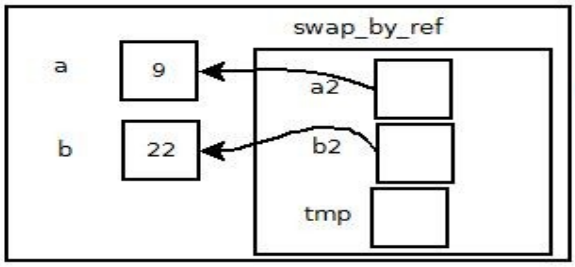
```
//This function swaps the content of two variables.
//@post This will change the contents of the variables that were
//    passed as arguments (in the scope of wherever this was
//    called.)
//@param a2 This value is a reference to the original
//    variable that was passed.
//@param b2 This value is a reference to the original
//    variable that was passed.
void swap_by_ref ( int& a2 , int& b2 )
{
      int tmp = a2;
      a2 = b2;
      b2 = tmp;
};

//This is meant to test the swap_by_ref() function.
//@post This will print the result of the test for the swap_by_ref()
function
//    to the console.
void testByRef()
{
      int a = 9;
      int b = 22;

      cout << "Before: a) " << a << " " << "b) " << b << endl;

      swap_by_ref(a,b);

      cout << "After : a) " << a << " " << "b) " << b << endl;
};
```

TestByReference



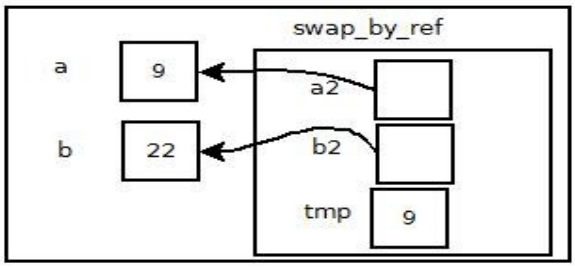
This is right before the swap_by_ref() function is called.

TestByReference



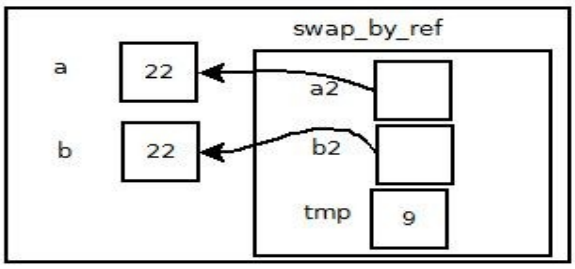
This is right after the swap_by_ref() function is called.

TestByReference



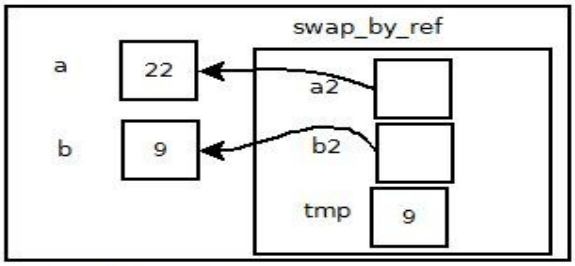
This is right after a2 is assigned to the tmp variable.

TestByReference



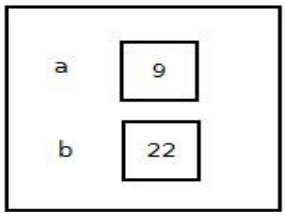
This is right after b2 is assigned to a2.

TestByReference



This is right after tmp is assigned to b2.

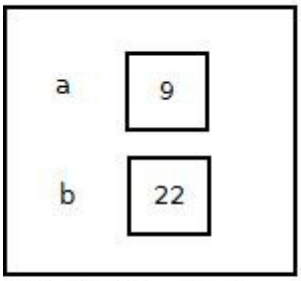
TestByReference



This is right after the swap_by_ref() function is called.

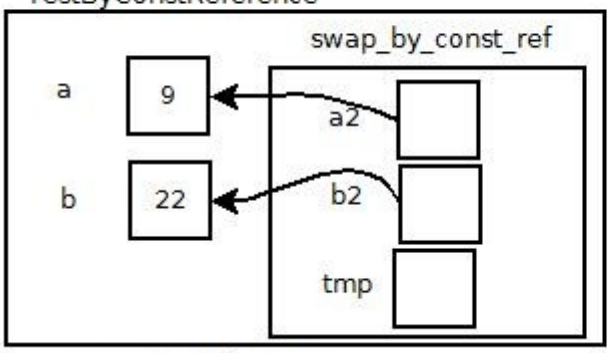
```
/*
//This function fails at swapping the content of two variables.
//@post This takes in constant references to some given
//    arguments, but is unable to swap the contents. That is
//    because constant variables can't be assigned to.
//@param a2 This value is a constant reference to the original
//    one that was passed.
//@param b2 This value is a constant reference to the original
//    one that was passed.
void swap_by_const_ref ( const int& a2 , const int& b2 )
{
     int tmp = a2;
     a2 = b2;
     b2 = tmp;
}
//This is meant to test the swap_by_const_ref() function.
//@post This will print the result of the test for the
//    swap_by_const_ref() function
//    to the console.
string testByConstRef()
{
     int a = 9;
     int b = 22;

     cout << "Before: a) " << a << " " << "b) " << b << endl;

     swap_by_const_ref(a,b);

     cout << "After : a) " << a << " " << "b) " << b << endl;
}
*/
```

TestByConstReference



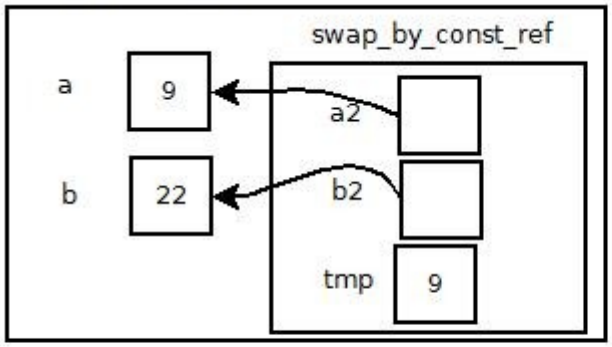
This is right before the swap_by_const_ref() function is called.

TestByConstReference



This is right after the swap_by_const_ref() function is called.

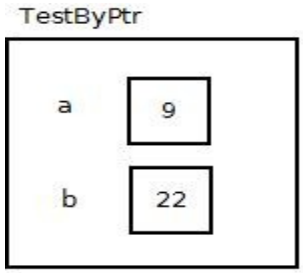
TestByConstReference



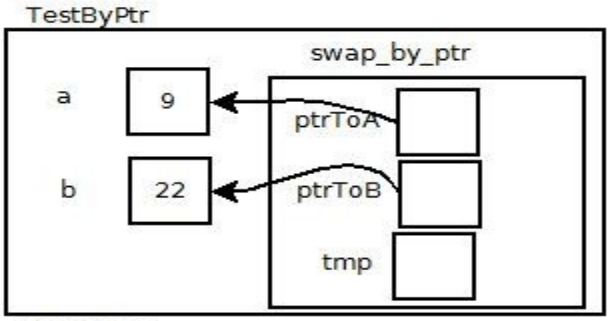
This is right after a2 is assigned to tmp

Then we can't go any further into the memory aspect of this stuff because a2 is constant and cannot be assigned b2.

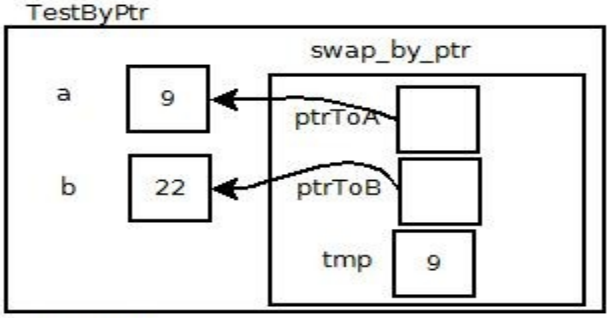
```
//This function swaps the content of two variables.
//@post This will change the contents of the variable addresses
//    that were passed as arguments (in the scope of wherever
//    this was called.)
//@param ptrToA This value is a pointer to the original
//    variable that was passed.
//@param ptrToB This value is a pointer to the original
//    variable that was passed.
void swap_by_ptr ( int* ptrToA, int* ptrToB )
{
      int tmp = *ptrToA;
      *ptrToA = *ptrToB;
      *ptrToB = tmp;
};

//This is meant to test the swap_by_ptr() function.
//@post This will print the result of the test for the
//    swap_by_ptr() function
//    to the console.
void testByPtr()
{
      int a = 9;
      int b = 22;

      cout << "Before: a) " << a << " " << "b) " << b << endl;

      swap_by_ptr(&a, &b);

      cout << "After : a) " << a << " " << "b) " << b << endl;
};
```



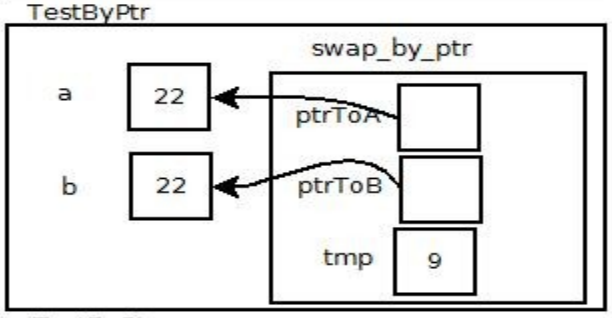
This is right before the swap_by_ptr() function is called.



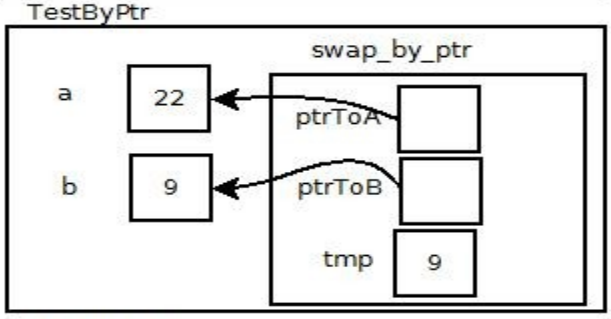
This is right after the swap_by_ptr() function is called.



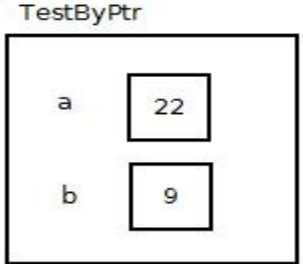
This is right after ptrToA is dereferenced and assigned to tmp.



This is right after ptrToB is dereferenced and assigned to the dereferenced ptrToA variable.

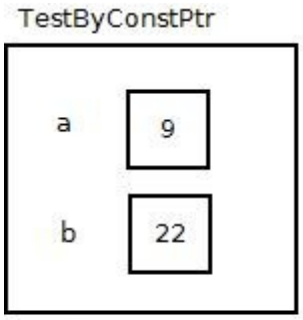


This is right after tmp is assigned to the dereferenced ptrToB variable.

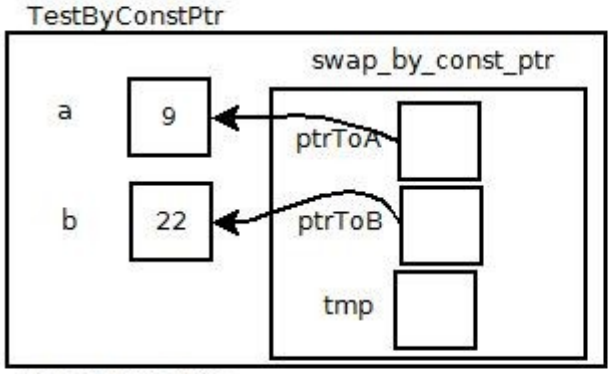


This is right after the swap_by_ptr() function is called.

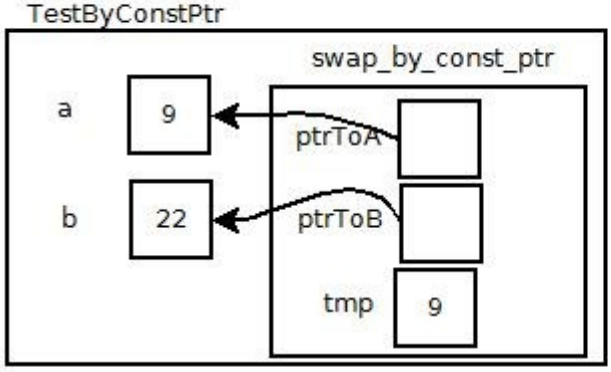
```
/*
//This function fails at swapping the content of two variables.
//@post This will throw an error because constant variables
//    cannot be assigned to (the const refers to the variable
//    being pointed to by the pointer.)
//@param ptrToA This value is a constant pointer to the original
//    variable that was passed.
//@param ptrToB This value is a constant pointer to the original
//    variable that was passed.
void swap_by_const_ptr ( const int* ptrToA , const int* ptrToB )
{
     int tmp = *ptrToA;
     *ptrToA = *ptrToB;
     *ptrToB = tmp;
};

//This is meant to test the swap_by_const_ptr() function.
//@post This will print the result of the test for the
//    swap_by_const_ptr() function
//    to the console.
void testByConstPtr()
{
     int a = 9;
     int b = 22;

     cout << "Before: a) " << a << " " << "b) " << b << endl;

     swap_by_const_ptr(&a, &b);

     cout << "After : a) " << a << " " << "b) " << b << endl;
};
*/
```



This is right before the swap_by_const_ptr() function is called.



This is right after the swap_by_const_ptr() function is called.



This is right after ptrToA is dereferenced and assigned to tmp.

Then we can't go any further with the memory aspect because the variable pointed to by ptrToA is flagged as constant. And the compiler is unable to assign values to a constant variable.

```
int main()
{
      //These are all of the functions that will execute.
      testByValue();
      testByRef();
      testByPtr();
      return 0;
};
```