Daniel Murga

Lab 9 Report

When implementing methods in C++, the ‘virtual’ modifier is added to the beginning of a function declaration to define it as virtual. In the compiled assembly code, the non-virtual method make_static_call has the ‘call _ZN5Class22static_dispatch_methodEv’ instruction, because the method contains the exact mangled name and address of the declared function. When calling the other two virtual methods, the call instruction is invoked on ‘*%eax’ which is a register containing the name of the virtual method, since it can be overloaded by different subclasses of the original base class and can not statically define a name. These two methods are able to call separate virtual functions because they use two different pointers to point to each unique virtual method. make_virtual_call uses the pointer offset by $8 and make_virtual_call_2 offset the location by $12. So while both of these functions are implemented in almost the same assembly code, the only difference is the pointer used to call the function.