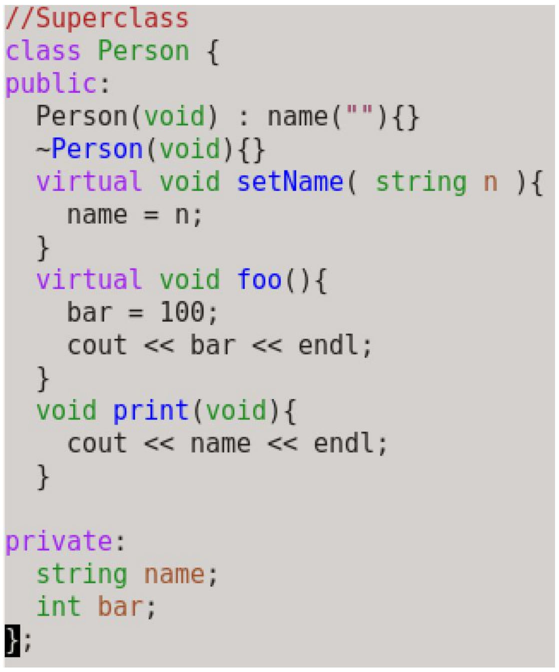
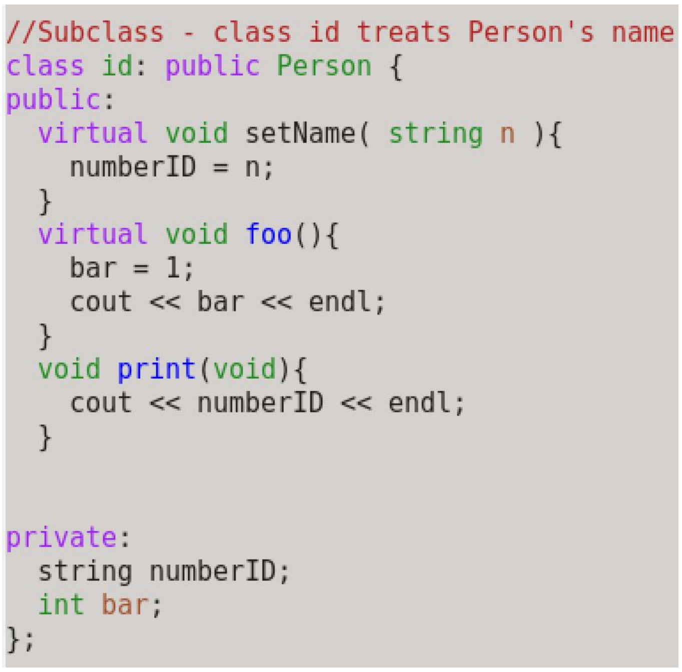
Ashley Nguyen - apn2my
InLab09 - inlab9.pdf
4/12/16

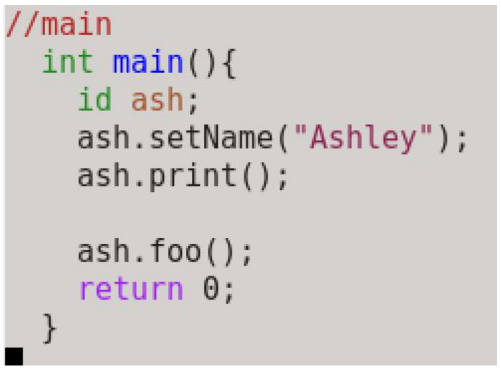
Dynamic Dispatch
When setting up fields of the superclass to be "inherited" by the subclass, dynamic dispatch is called when the keyword virtual is used. If the keyword virtual is preceded by the methods of a certain field (i.e. public, private, protected) then the compiler will check to see if the subclass redefines this method at run time. If the subclass does indeed redefine this method then the subclass's methods will be the ones that are used to run the program.



```
//Superclass
class Person {
public:
  Person(void) : name(""){}
  ~Person(void){}
  virtual void setName( string n ){
    name = n;
  }
  virtual void foo(){
    bar = 100;
    cout << bar << endl;
  }
  void print(void){
    cout << name << endl;
  }

private:
  string name;
  int bar;
};
```

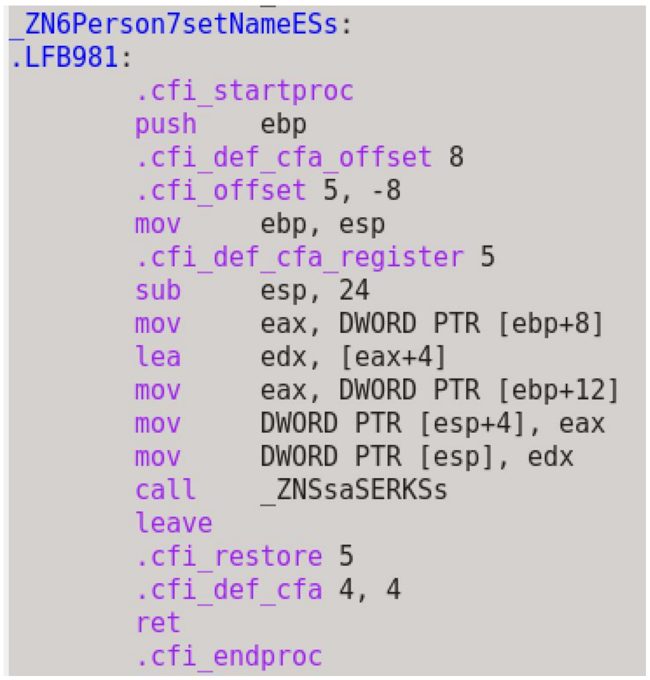


```
//Subclass - class id treats Person's name
class id: public Person {
public:
  virtual void setName( string n ){
    numberID = n;
  }
  virtual void foo(){
    bar = 1;
    cout << bar << endl;
  }
  void print(void){
    cout << numberID << endl;
  }


private:
  string numberID;
  int bar;
};
```

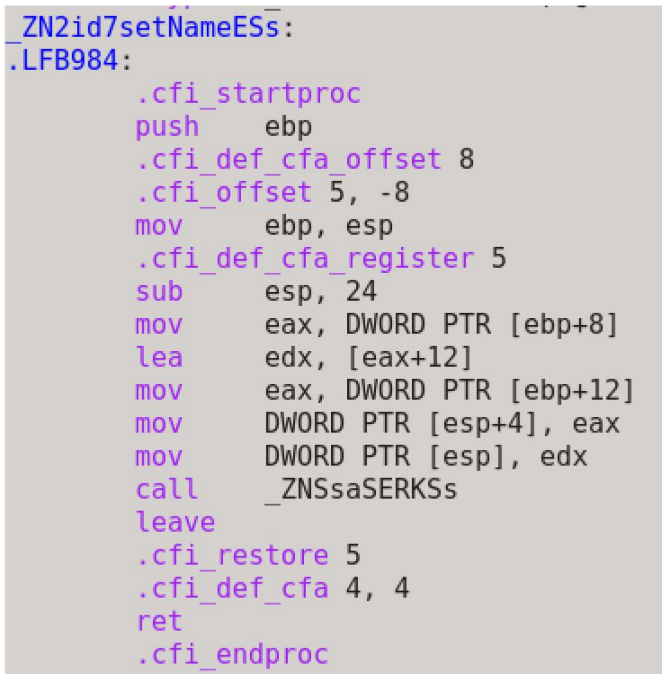


```
//main
  int main(){
    id ash;
    ash.setName("Ashley");
    ash.print();

    ash.foo();
    return 0;
  }
```

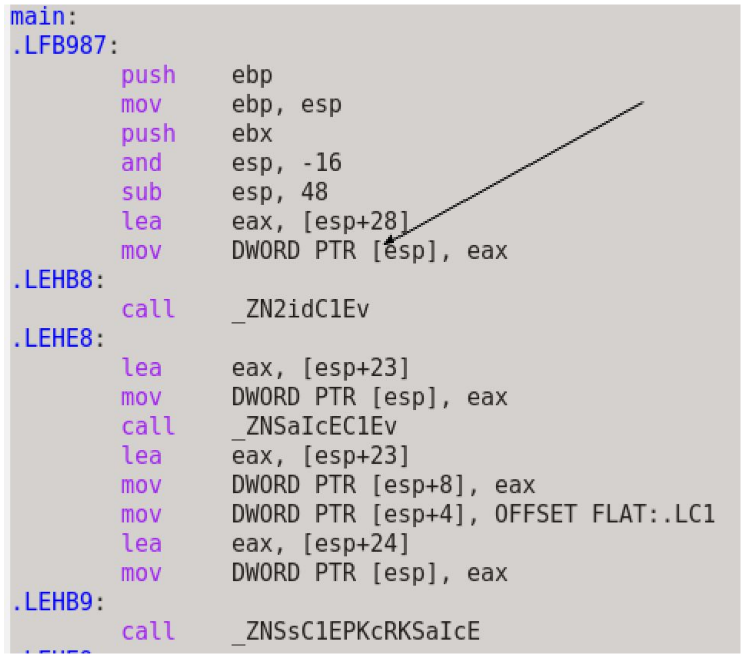
The Person class is the superclass and the id class is the subclass. These two classes have two methods that are identical, the setName method takes in a string parameter and the foo() method. The main method shows the ambiguity in which setName() and which foo() method the id class will inherit. Looking at the assembly may provide answers.



```
_ZN6Person7setNameESs:
.LFB981:
        .cfi_startproc
        push    ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        mov     ebp, esp
        .cfi_def_cfa_register 5
        sub     esp, 24
        mov     eax, DWORD PTR [ebp+8]
        lea     edx, [eax+4]
        mov     eax, DWORD PTR [ebp+12]
        mov     DWORD PTR [esp+4], eax
        mov     DWORD PTR [esp], edx
        call    _ZNSsaSERKSs
        leave
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
```



```
_ZN2id7setNameESs:
.LFB984:
        .cfi_startproc
        push    ebp
        .cfi_def_cfa_offset 8
        .cfi_offset 5, -8
        mov     ebp, esp
        .cfi_def_cfa_register 5
        sub     esp, 24
        mov     eax, DWORD PTR [ebp+8]
        lea     edx, [eax+12]
        mov     eax, DWORD PTR [ebp+12]
        mov     DWORD PTR [esp+4], eax
        mov     DWORD PTR [esp], edx
        call    _ZNSsaSERKSs
        leave
        .cfi_restore 5
        .cfi_def_cfa 4, 4
        ret
        .cfi_endproc
```



```
main:
.LFB987:
        push    ebp
        mov     ebp, esp
        push    ebx
        and     esp, -16
        sub     esp, 48
        lea     eax, [esp+28]
        mov     DWORD PTR [esp], eax
.LEHB8:
        call    _ZN2idC1Ev
.LEHE8:
        lea     eax, [esp+23]
        mov     DWORD PTR [esp], eax
        call    _ZNSaIcEC1Ev
        lea     eax, [esp+23]
        mov     DWORD PTR [esp+8], eax
        mov     DWORD PTR [esp+4], OFFSET FLAT:.LC1
        lea     eax, [esp+24]
        mov     DWORD PTR [esp], eax
.LEHB9:
        call    _ZNSsC1EPKcRKSaIcE
```

As shown in the Person class and id class assembly of the setName() method, the assembly code is similar at the machine level. However, at runtime in the main method, under .LEHB8 header, the id class is called when calling the setName() method. This shows that using the virtual keyword will enable dynamic dispatch so that the compiler knows to use the subclass's redefined method instead of the superclass's method. Thus in the Virtual Method Table, the value stored in that portion of the stack will reference the memory address of the subclass rather than the superclass. Similar findings would also be shown for the identical foo() method as well.