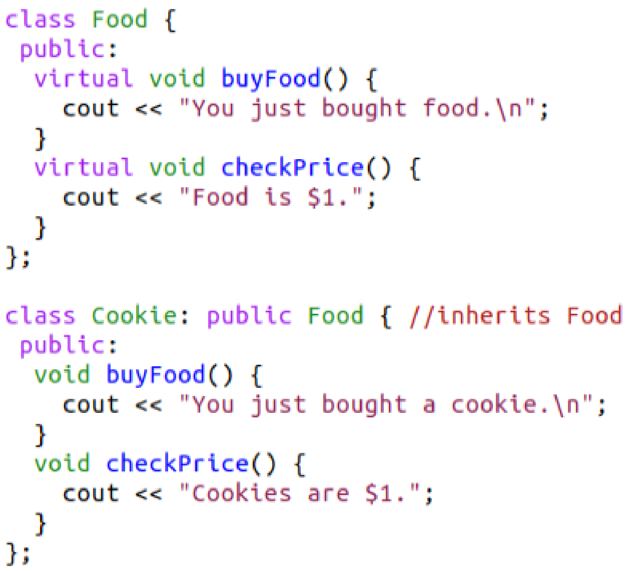
Samreen Azam (sa3tnc)
11/19/19
inlab9.pdf

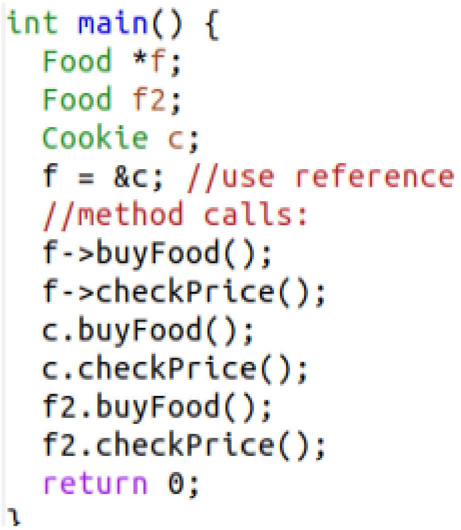
Dynamic Dispatch:

Dynamic dispatch refers to the process of determining which version of a polymorphic function should be called during runtime[1]. To see how this is implemented, I created two simple classes: Food and Cookie, which inherits Food. Their implementations in C++ are shown below.



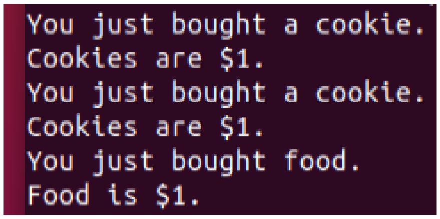
```
class Food {
 public:
  virtual void buyFood() {
    cout << "You just bought food.\n";
  }
  virtual void checkPrice() {
    cout << "Food is $1.";
  }
};

class Cookie: public Food { //inherits Food
 public:
  void buyFood() {
    cout << "You just bought a cookie.\n";
  }
  void checkPrice() {
    cout << "Cookies are $1.";
  }
};
```



```
int main() {
  Food *f;
  Food f2;
  Cookie c;
  f = &c; //use reference
  //method calls:
  f->buyFood();
  f->checkPrice();
  c.buyFood();
  c.checkPrice();
  f2.buyFood();
  f2.checkPrice();
  return 0;
}
```

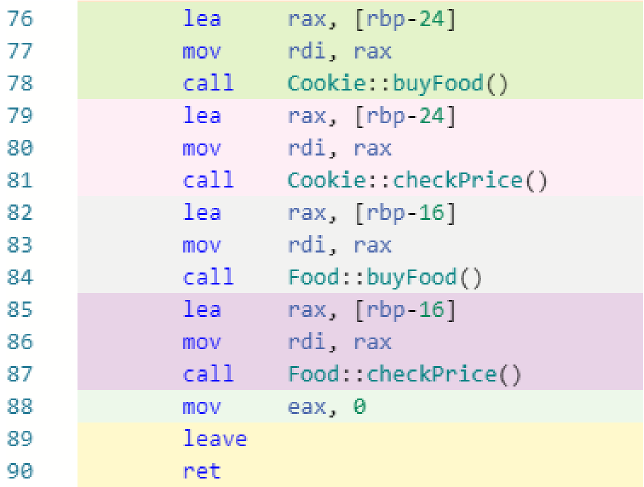
Output:



```
You just bought a cookie.
Cookies are $1.
You just bought a cookie.
Cookies are $1.
You just bought food.
Food is $1.
```

This occurs because f is set to point to c, and thus can use Cookie's buyFood() and checkPrice() methods. For c, the overridden methods in its class are called. Finally, f2 only has access to the operations in the Food class, so it can only call the original virtual methods.

Next, I generated assembly code via godbolt.com. The following image depicts the function calls in main:



```
76          lea     rax, [rbp-24]
77          mov     rdi, rax
78          call    Cookie::buyFood()
79          lea     rax, [rbp-24]
80          mov     rdi, rax
81          call    Cookie::checkPrice()
82          lea     rax, [rbp-16]
83          mov     rdi, rax
84          call    Food::buyFood()
85          lea     rax, [rbp-16]
86          mov     rdi, rax
87          call    Food::checkPrice()
88          mov     eax, 0
89          leave
90          ret
```

The methods in the Food class are only called at the end, at Lines 84 and 87. On the other hand, the Cookie class methods are called first. This conveys how only f2 uses the methods in the Food class whereas c and f call Cookie's methods. Since f is a reference type that holds the address of c, this is expected.

[1] Source: https://condor.depaul.edu/ichu/csc447/notes/wk10/Dynamic2.htm