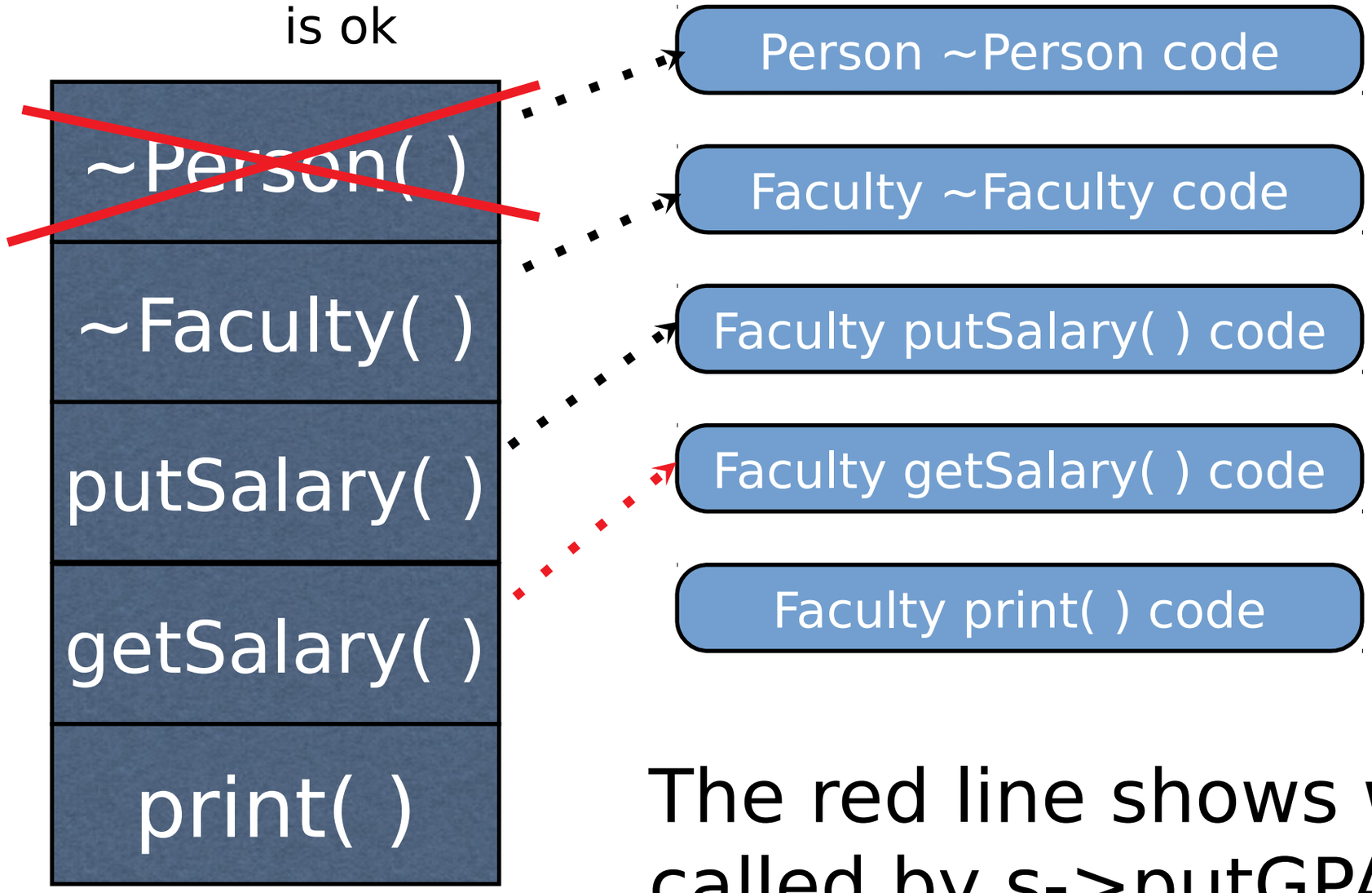
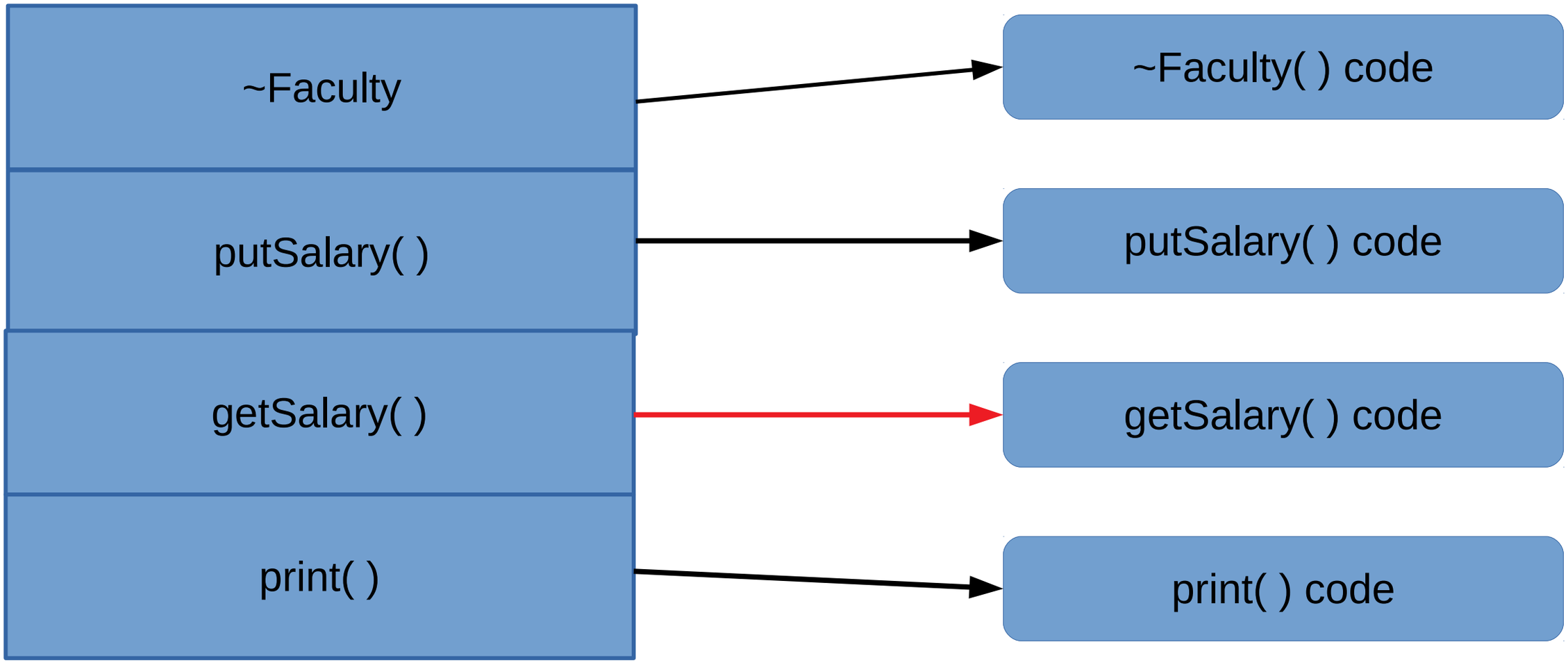
destructors are optional, hand drawn is ok

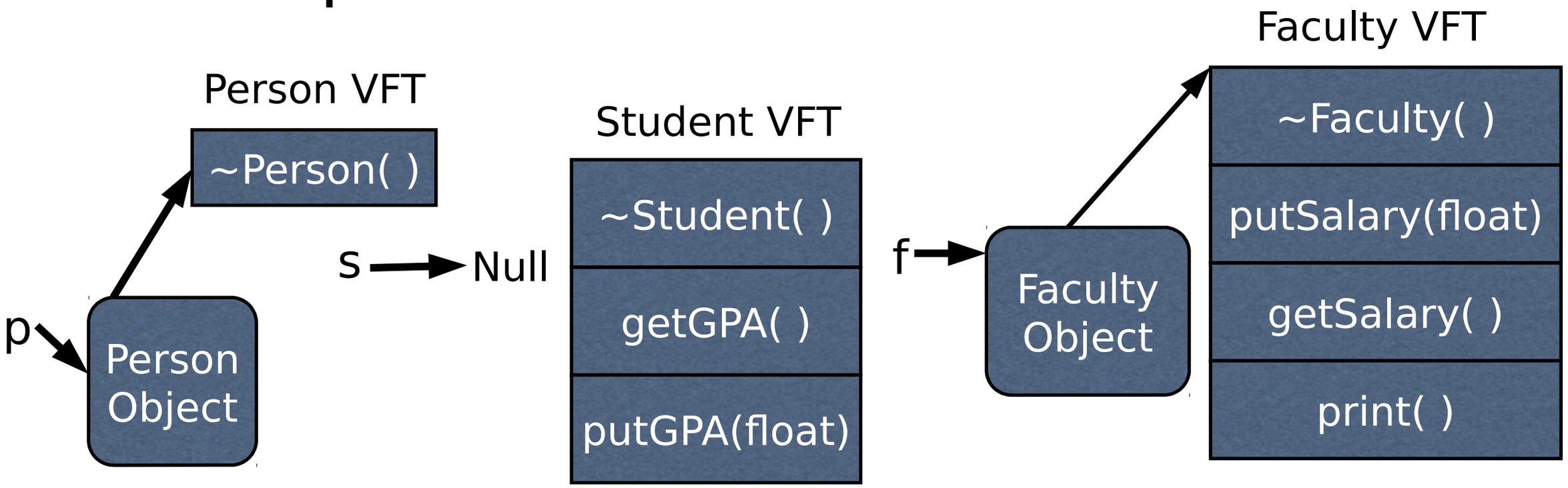


The red line shows what is called by s->putGPA(6,6)



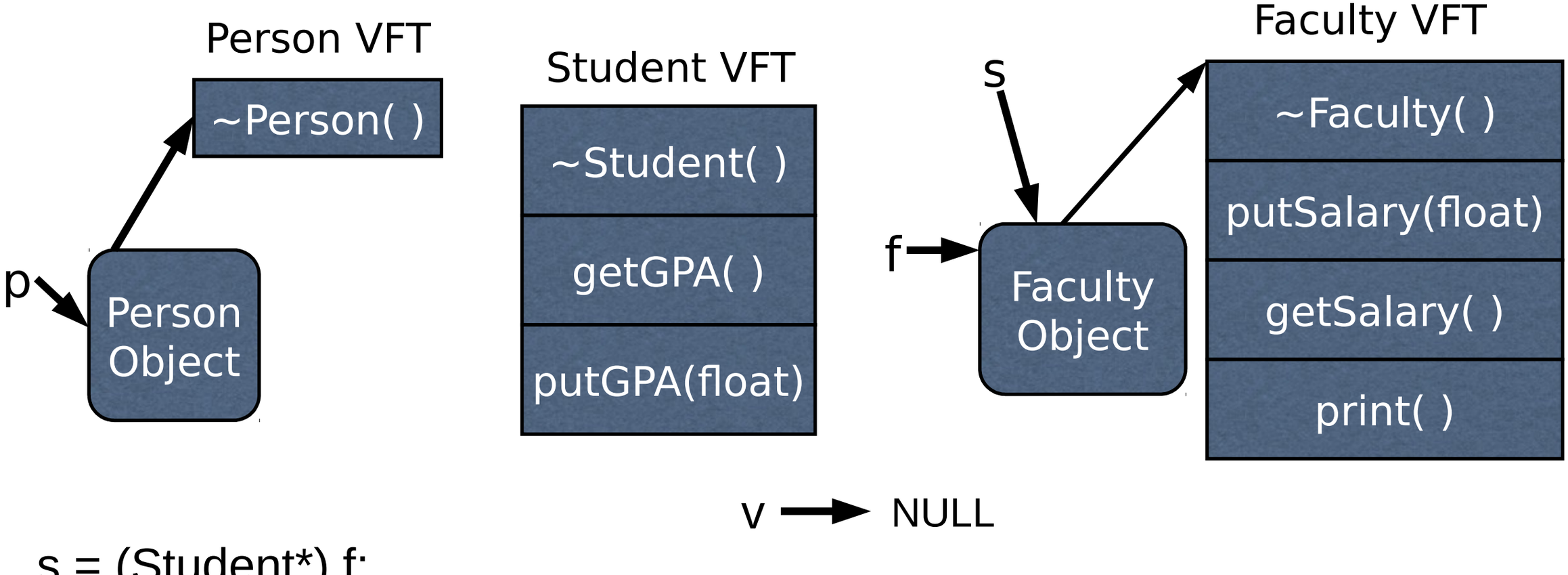
The red arrow shows what is called by *s->putGPA(6,6)*

Group 0



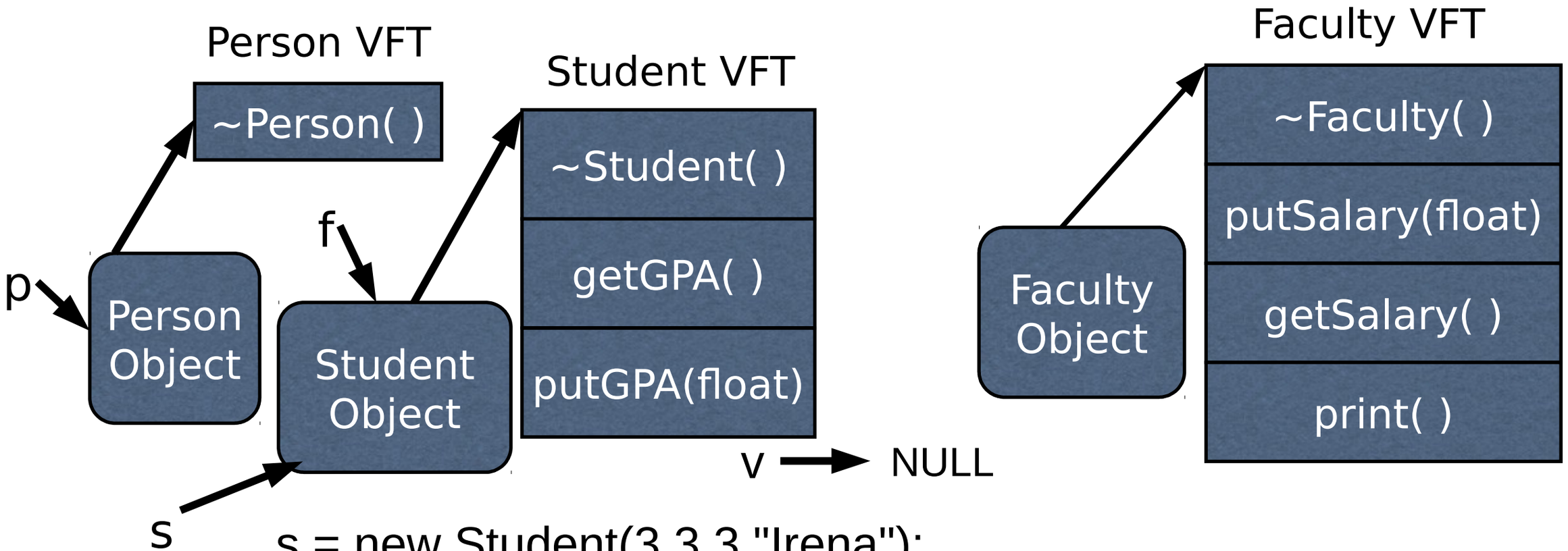
v → NULL

```
Person* p = new
Person(1,1,1,"Bob");
Faculty* f = new
Faculty(2,2,2,"Dumbledore");
Student* s;
void* v;
```



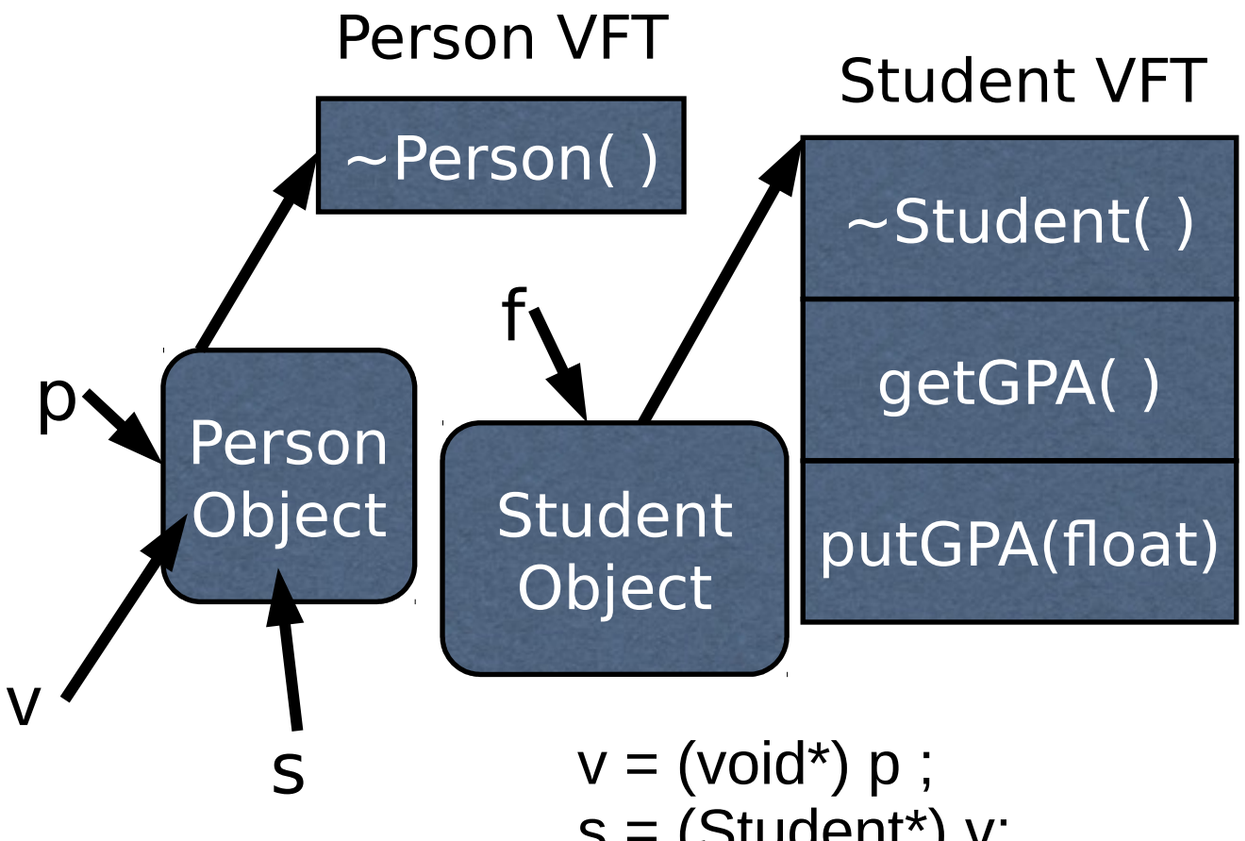

```
s = (Student*) f;
std::cout << "before (s=f).putGPA(6,6)" << std::endl;
s->putGPA(6.6);
std::cout << "before (s=f).getGPA" << std::endl;
s->getGPA( );
```

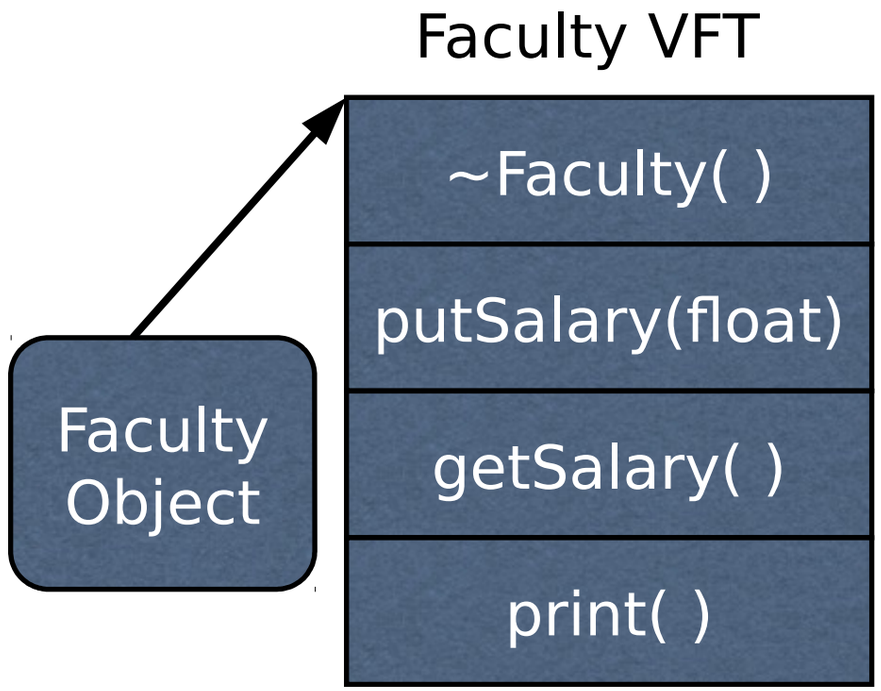
When s->puGPA(6.6) is called, the Student class is checked for a matching function, and putGPA(float) is found. It is in the 3rd VFT slot. Since the method is virtual and the function is called through a pointer the VFT of the object pointed to is used to call the function, resulting in the Faculty getSalary function being called. Similar things happen with getGPA, and putSalary(float) is called.




```
s = new Student(3,3,3,"Irena");
f = (Faculty *) s; // compiles, but will cause problems at runtime
std::cout << "before (f=s).putSalary" << std::endl;
f->putSalary(500.0);
std::cout << "before (f=s).getSalary" << std::endl;
f->getSalary( );
```

When f->putSalary(500.0) is called, the Faculty class is checked for a matching function, and putSalary(float) is found. It is in the 2nd VFT slot. Since the method is virtual and the function is called through a pointer the VFT of the object pointed to is used to call the function, resulting in the Student getGPA( ) function being called. Similar things happen with getSalary( ) and putGPA(float) is called.






```
v = (void*) p ;
s = (Student*) v;
std::cout << "before (s=p).getGPA" << std::endl;
s->getGPA( );
std::cout << "finished" << std::endl;
```

The cast and assignment into the void pointer v, and then cast and assignment of v into s result in a Student pointer s pointing to a Person object. getGPA( ) is called. This is matched to the Student getGPA( ) (because s is a pointer to a Student), which is in the 2nd VFT slot. The 2nd (and non-existent) slot of of the Person VFT is accessed to call the getGPA( ) function. Whatever is in memory at this location is treated as the function address, and a call is made to that address. On my system I got a segmentation fault.

What can we learn from this?

- Casts should be treated with respect, especially C-style casts
- When calling a virtual function, only the type of the base pointer or reference counts in determining which VFT slot to call through – the type of the object pointed to is irrelevant
- Just because your program compiles doesn’t mean that it is a standard conforming program. From https://en.cppreference.com/w/cpp/language/ub

The C++ standard precisely defines the observable behavior of every C++ program that does not fall into one of the following classes:

- undefined behavior - there are no restrictions on the behavior of the program. Examples of undefined behavior are memory accesses outside of array bounds, signed integer overflow, null pointer dereference, modification of the same scalar more than once in an expression without sequence points, access to an object through a pointer of a different type, etc. Compilers are not required to diagnose undefined behavior (although many simple situations are diagnosed), and the compiled program is not required to do anything meaningful.