Employee

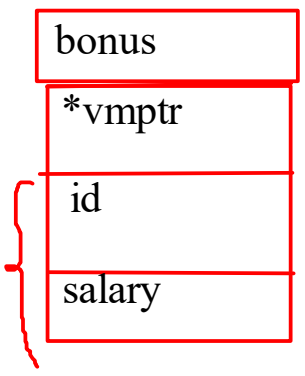
Manager Salesman

SalesManager

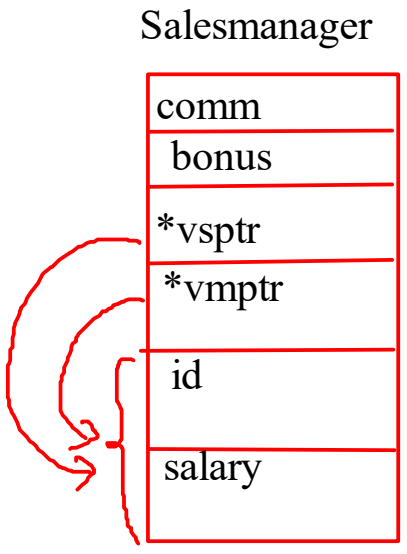
Employee *e = new Salesman();

Manager *m = (Manager *) e;

dynamic_cast



Salesmanager



STL - Standard Template Library
1. Containers
2. Algorithms
3. Function Objects
4. Iterators

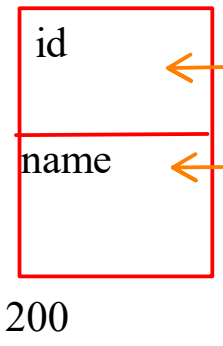
1. Sequence Containers
2. Associate Containers
3. Sequence Container Adapters

vector
map
stack,queue

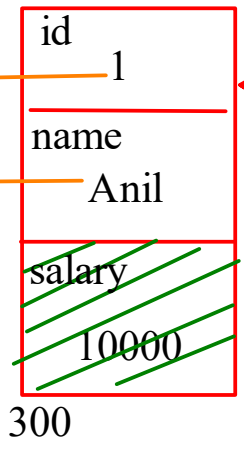
vector<Person*> list;

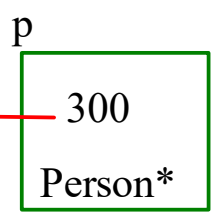
Employee e2= p;

Person p;



Employee e;





Custome -> multiple products

p=e;

```
class Person{
name
}

class Customer : public Person{
mobile
vector<Product> productList;

addProduct(){
for()
        Product *p = new Product();
        p->accept();
        productList.push_back(p);
}

vector<Product*>& getProductList(){
        return productList;
}
}
```

```
class Product{
id=1,
pname,
price

}
```

vector<Product> products;

```
vector<Customer*> customerList;
Customer *c;
for(){
        c = customerList[i];
        if(mobile == customerList[i]->getMobile())
                customerList[i]->addProduct();


                Product *p = new Product();
                p->accept();
                customerList[i]->getProductList().push_back(p);
}
```

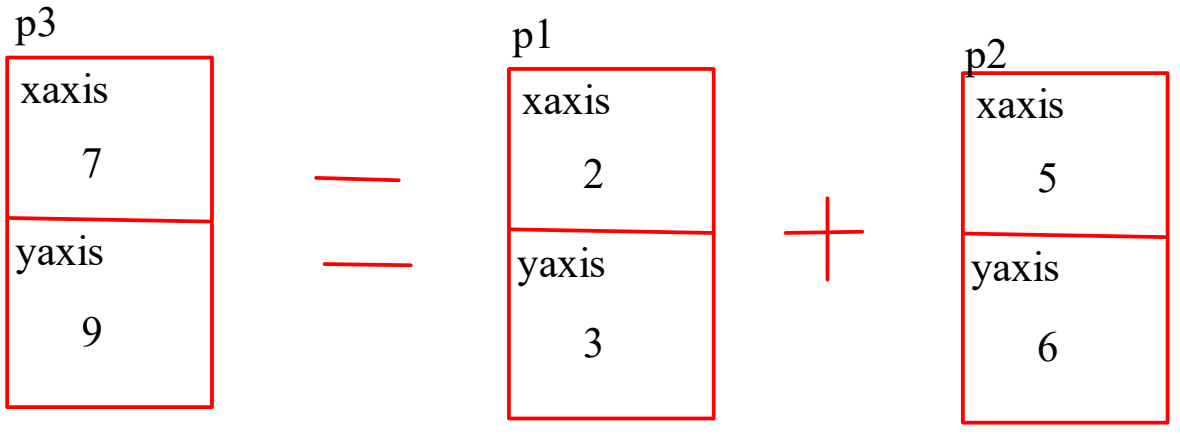
```
int find(vector<Person*>& list, int flag){
for(int i;<list.size;i++)
        if(flag == 0) // student
                if(typeid(*list[i])==typeid(Student))
                        if(id==list[i]->getId())
                                return i;
        elseif(flag ==1) // employee
                if(typeid(*list[i])==typeid(Employee))
                        if(id==list[i]->getId())
                                return i;
return -1;
}
```

```
case 5:
int index = find(personList,1);
break;
case 6:
int index = find(personList,0);
```

```
+
-
<<
>>
=
[]
()
->
```

1 , 2 5, 6
3, 4 + 7, 8

m1 m2 m1+m2

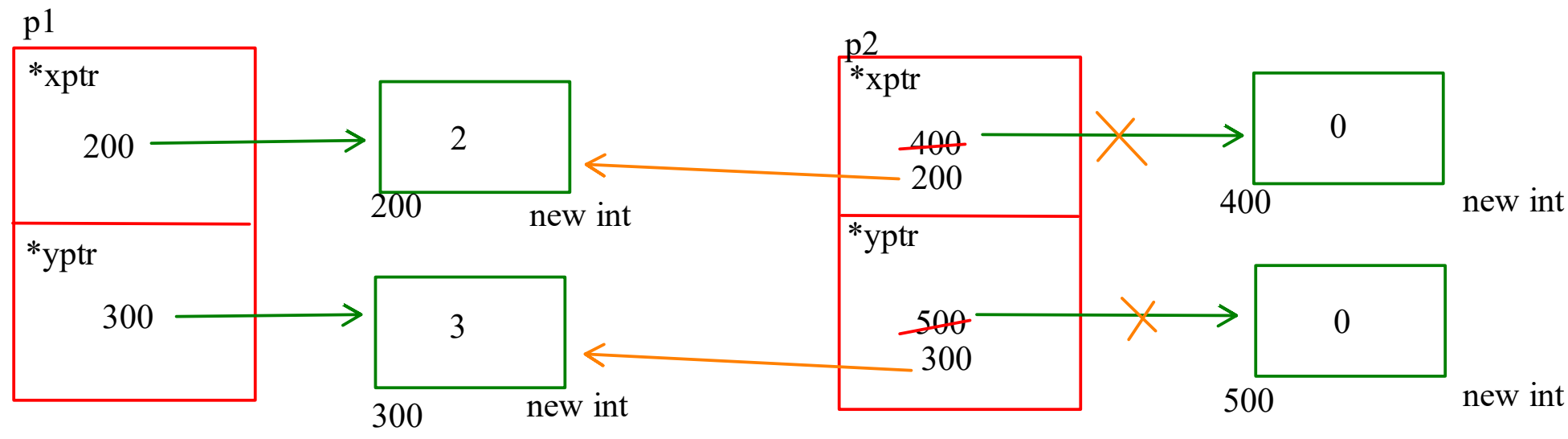


p3.xaxis = p1.xaxis + p2.xaxis;
= 2 + 5
= 7

>

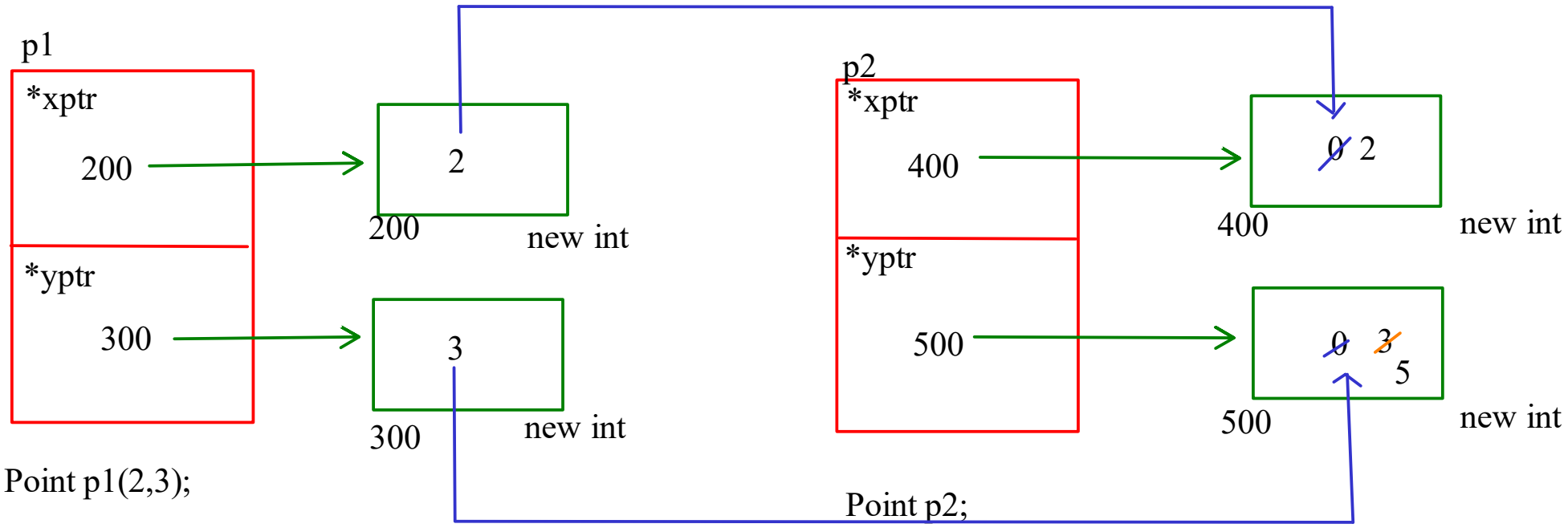
Point p1(2,4);
Point p2 = (2,4);// copy ctor

Point p1(2,4);
Point p2; // parameterless
p2 = p1; // Assignment operator function



Point p1(2,3);

Point p2;



p2 = p1;

*p2.xptr = *p1.xptr;
*p2.yptr = *p1.yptr;