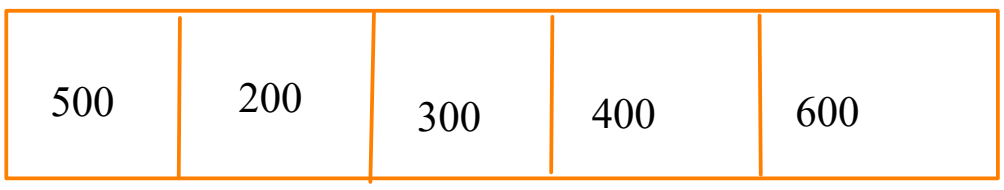
```
Enter the rollno -
cin>>rn
for()
        if(arr[i]->getRollno()==rollno)
                arr[i]->display();
```

Student*arr[5]



Hirerachy
- has-a
- is-a

```
arr[i]->getRollno()
>
arr[j]->getRollno()
```

200

```
Student *temp = arr[i];
arr[i]=arr[j];
arr[j]=temp
```

```
class Date{
}

class Employee : Person{
Date doj;
Date dob;
Date terDate;
 }
```

```
class Person{
string name;
string mobile;
}
```

Association
- Composition -> Date d
- Aggegration -> Date *d

Inheritance -> Base/Parent -> Derived/Child

```
class Base{

}

class Derived : Base{

}
```

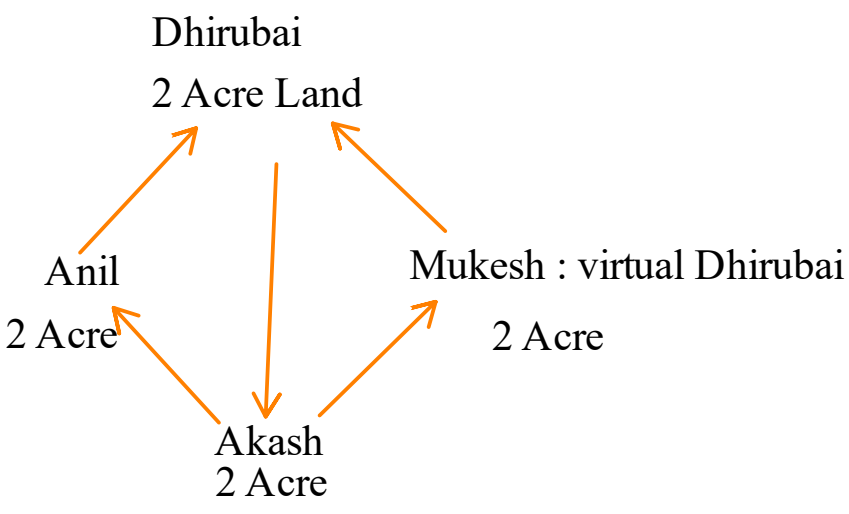
1. Ctor
2. Dtor
3. Copy ctor
4. Assignment Operator function
5. Friend Function

A

B C

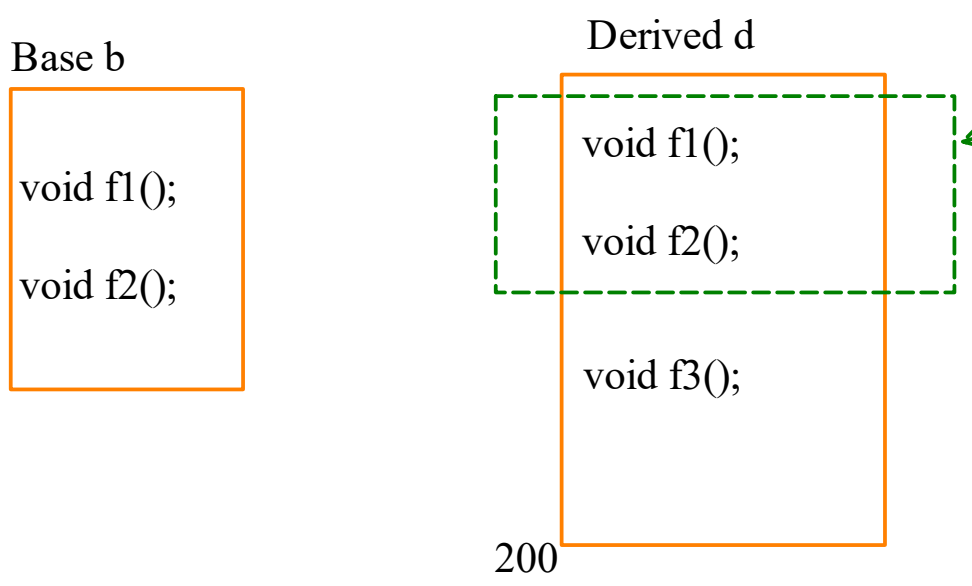
D

Diamond Problem
- Members of Indirect Base class inheriting multiple times into Indirect Derived class represents Diamond Problem



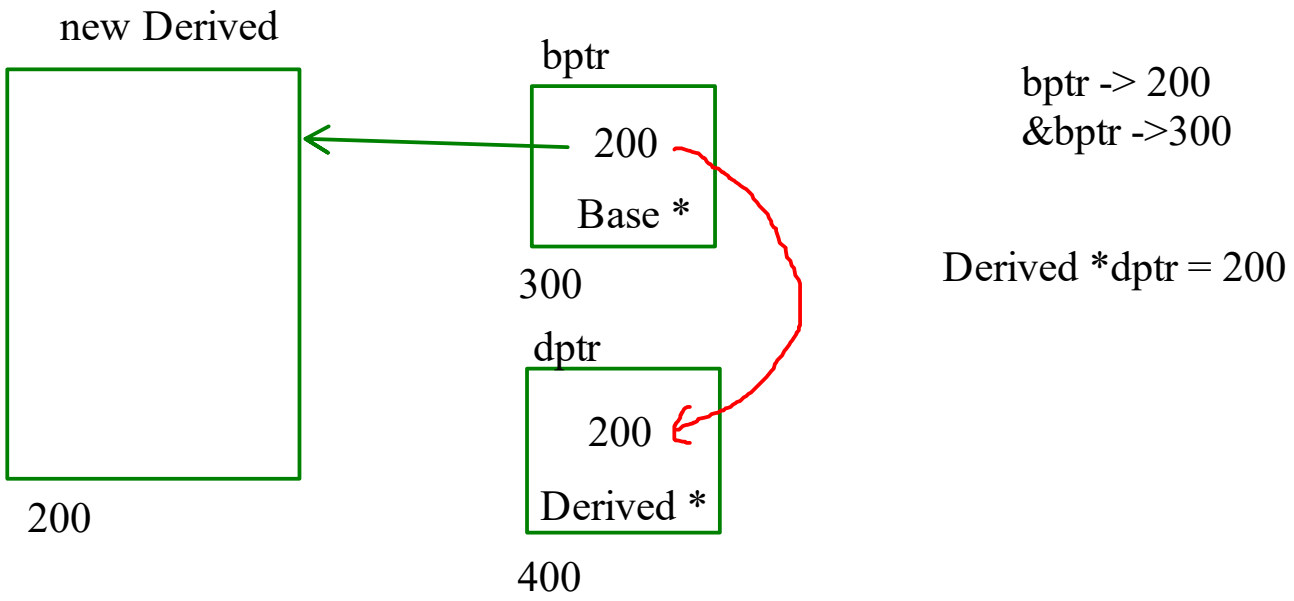

```
A()
B()
C()
D()
```

##Upcasting
- Keeping the object od derived classs into the base class pointer or reference is upcasting



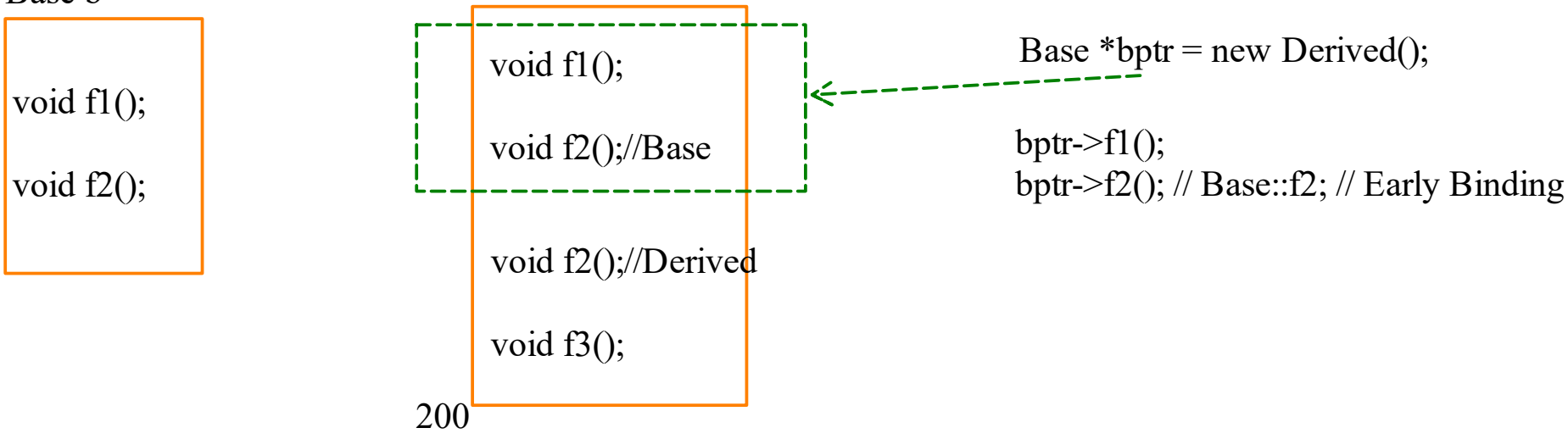

```
Base *bptr = new Derived();

bptr->f1();
bptr->f2();

Derived *dptr =(Derived *) bptr; // Downcasti
f3();
```

Downcasting
- Process of convverting base class pointer/reference to dervied class pointer/refrence is called downcasting
- At the time of downcasting explicit typecasting is mandatory



Function Overriding

- Redefining the base class function once again inside the derived class with same name and signature is called as function overriding
- function overriding is an example of Run time polymorphism



Early Binding

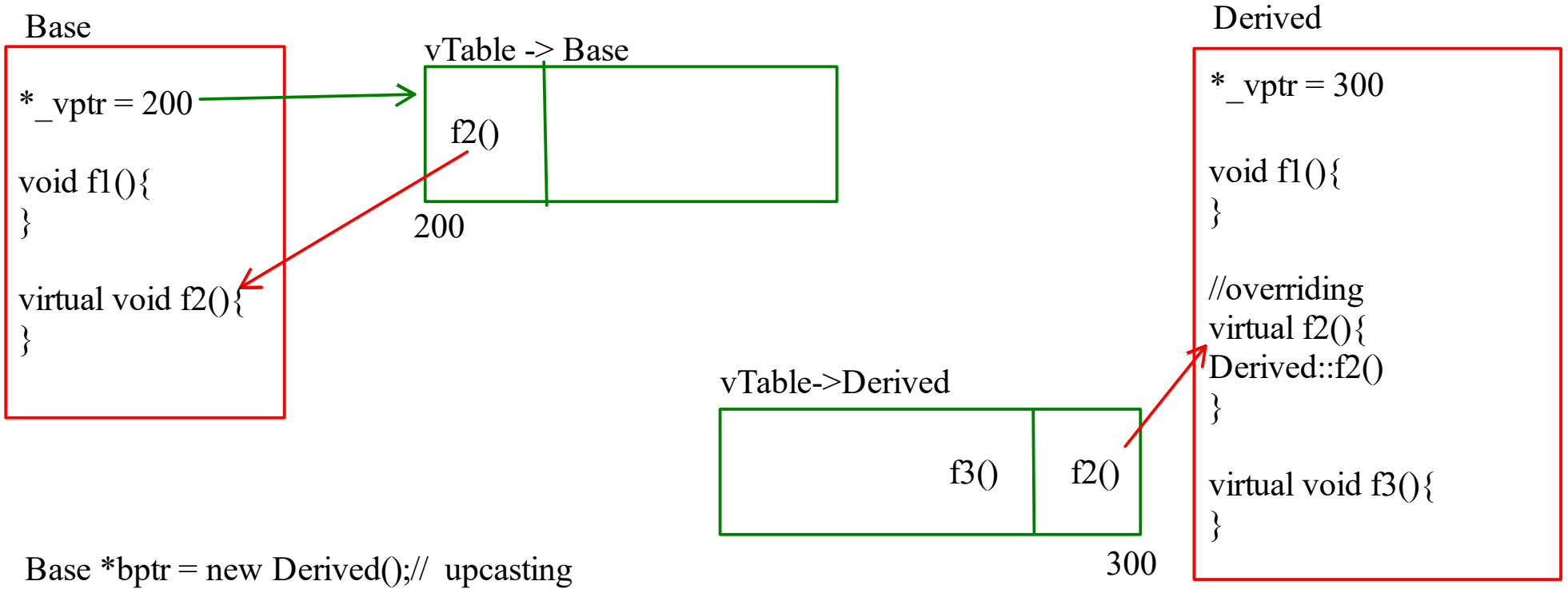
- A call to a function is resolved based on the type of pointer , it is called as Early Binding

Late Binding

- A call to a function is resolved based on object that is created then its is called as Late Binding

Purpose of overriding

- Base class member function is 100% incomplete
- Base class member function is partial complete
- Different implementation from the base class function




```
Base *bptr = new Derived();//  upcasting
bptr->f1(); // Base
bptr->f2();// Derived::f2
```

Shape

Circle is-a Shape

Rectangle is-a Shape

RTTI -> Runtime Type Information

type_info

```
int main(){
Employee e1()

e1.setId();
}
```

Virtual Destructor

menu()
0. exit
1. Add Employee
2. Add Student
3. Display All Employees
4. Display All Students
5. Find Student
6. Find Employee