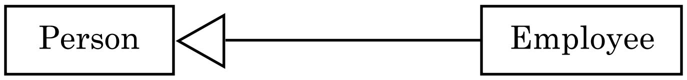
Mapping class diagram relationships to implementation

01. Inheritance

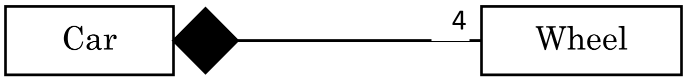


```
class Person
{
        protected:
                <attributes>
        public:
                <methods>
};
```

```
class Employee : public Person
{
        private:
                <attributes>
        public:
                <methods>
};
```

---

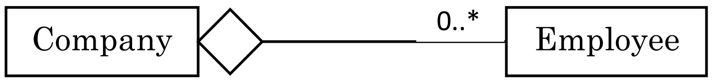
02. Composition



```
class Car
{
        private:
                <attributes>
                Wheel * wheels [4];
        public:
                <methods>
                void addWheels()
                {
                        Wheels [0] = new Wheel();
                        Wheels [1] = new Wheel();
                        Wheels [2] = new Wheel();
                        Wheels [3] = new Wheel();
                }
};
```

```
class Wheel
{
        private:
                <attributes>
        public:
                <methods>
};
```

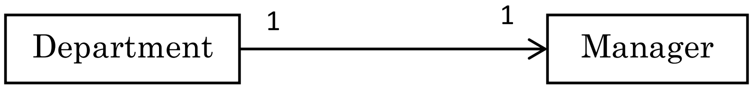
03. Aggregation



```
#define SIZE 2
class Company
{
        private:
                <attributes>
                Employee * employees [SIZE];
        public:
                <methods>
                void addEmployees(Employee *
                                e1, Employee * e2)
                {
                        employees [0] = e1;
                        employees [1] = e2;
                }
};
```

```
class Employee
{
        private:
                <attributes>
        public:
                <methods>
};
```

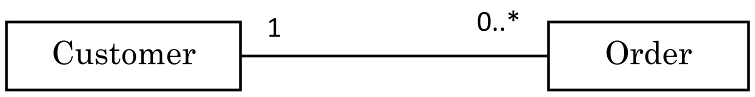
04. Uni-directional Association



```
class Department
{
        private:
                <attributes>
                Manager * mgr;
        public:
                <methods>
};
```

```
class Manager
{
        private:
                <attributes>
        public:
                <methods>
};
```

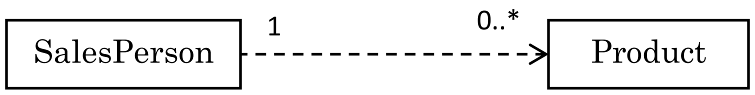
05. Bi-directional Association



```
#define SIZE 10
class Customer
{
        private:
                <attributes>
                Order * orders[SIZE];
        public:
                <methods>
};
```

```
class Order
{
        private:
        <attributes>
                Customer * cus;
        public:
                <methods>
};
```

06.Dependancy



```
class SalesPerson
{
        private:
                <attributes>
        public:
                <methods>
                void addSales(int qty , Product *P);
};
```

```
class Product
{
        private:
                <attributes>
        public:
                <methods>
};
```