

ШИНЖЛЭХ УХААН ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
МЭДЭЭЛЭЛ, ХОЛБООНЫ ТЕХНОЛОГИЙН СУРГУУЛЬ

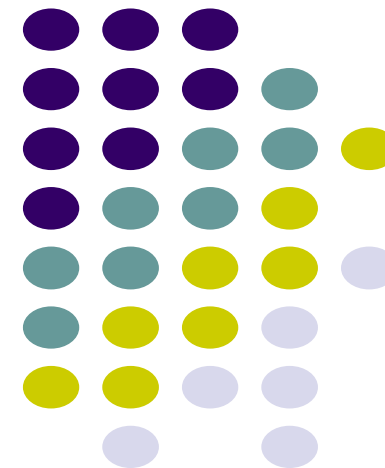
Ү.СS202 ОБЪЕКТ ХАНДЛАГАТ ПРОГРАМЧЛАЛ

Лекц №6

Бүрдмэл харьцаа

док., дэд проф. Б.Батзолбоо
маг. Б.Мөнхбуян

2021 он



Агуулга



- Бүрдмэл харьцаа
- Бүрдмэл харьцааны хэлбэр
- Гишүүн өгөгдөл хэлбэрээр хэрэглэх
- Параметр хэлбэрээр хэрэглэх
- Буцаах төрөл хэлбэрээр хэрэглэх
- Объектын харилцан үйлчлэл болон бүрдмэл харьцаа

Өмнөх лекцээр

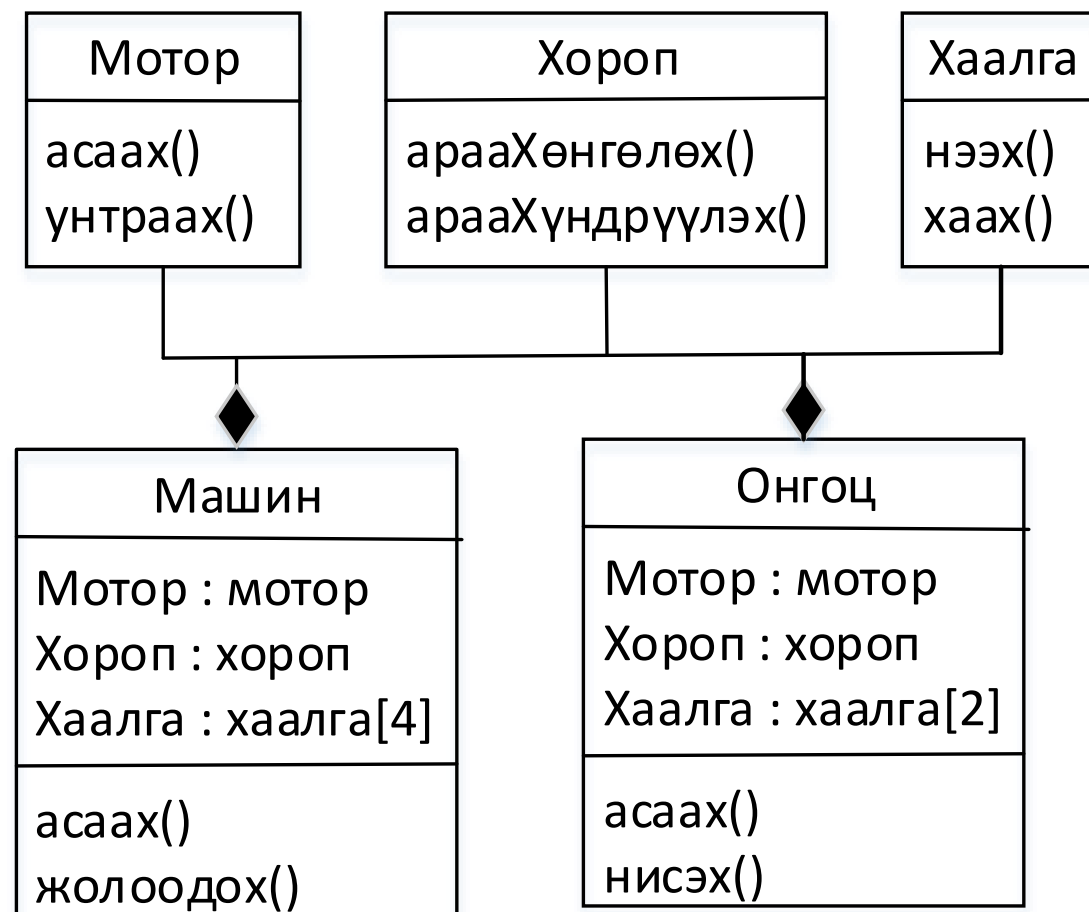


- Жавагийн стандарт классын сангуудтай танилцсан
 - Math
 - String
 - Date
- Объектыг хэрхэн харьцуулахыг сурсан

Бүрдмэл харьцаа



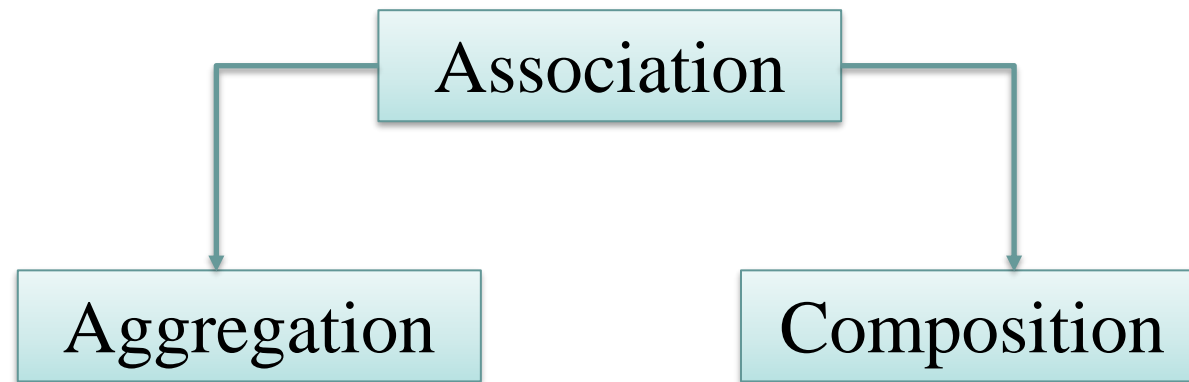
- Өмнөх хичээлд нэг классын тухай үзэж байсан бол одоо классуудын харилцан үйлчлэлийн хамгийн энгийн хэлбэрийг үзэж байна.
- Гол санаа нь байгаа объектуудыг хослуулан шинэ объект үүсгэх юм. Энэ нь дахин ашиглагдах хэлбэр юм.





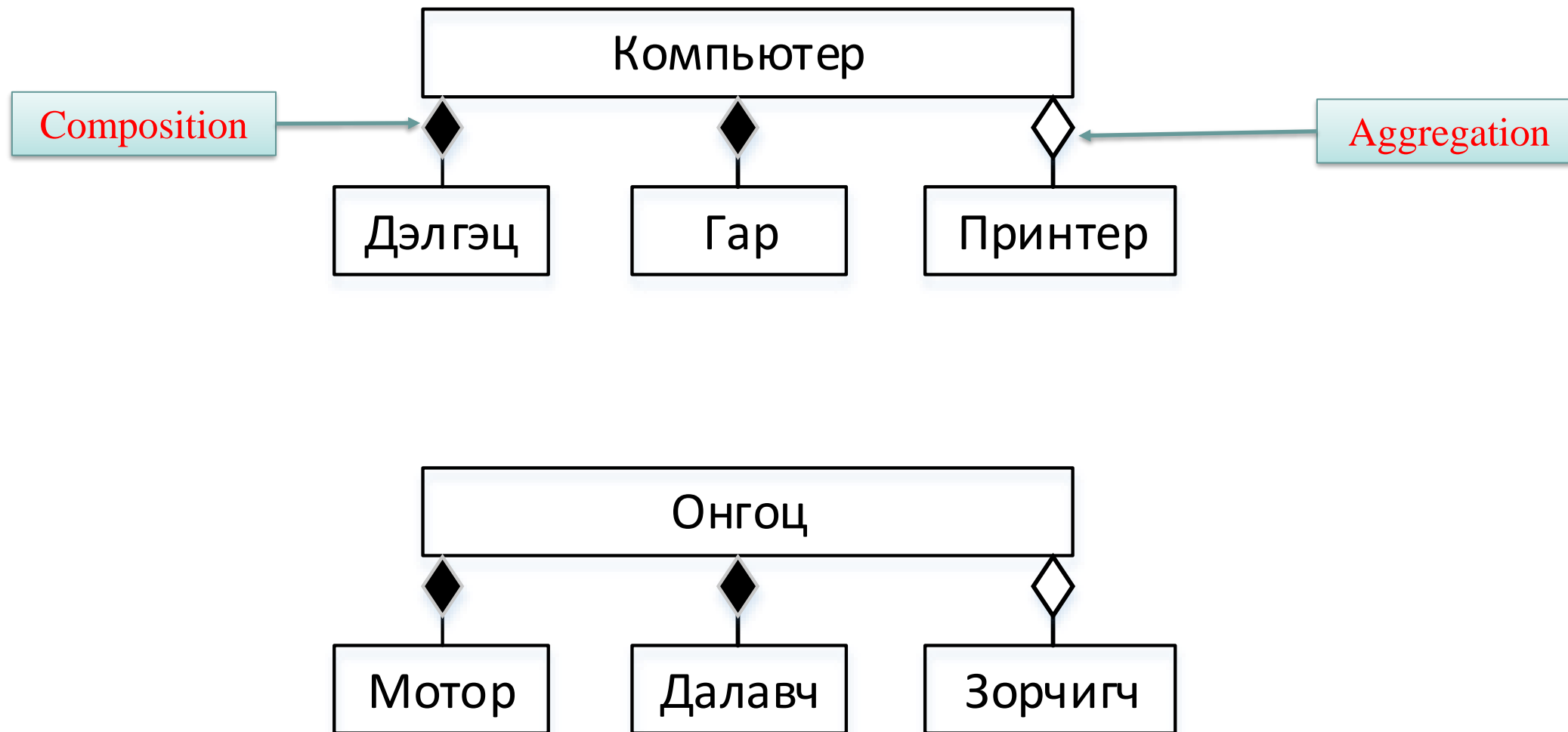
Бүрдмэл харьцаа

- Объектуудын хоорондын харилцааг association гэх бөгөөд нэг объект нөгөөхөө эзэмшиж байвал composition, харин нэг объект нөгөөгөө ашиглаж байвал aggregation гэнэ.



- Бүтэцлэг / Composition – хатуу холбоо, заавал байх
- Нийлмэл / Aggregation – сул холбоо, заавал байх албагүй, байж болох

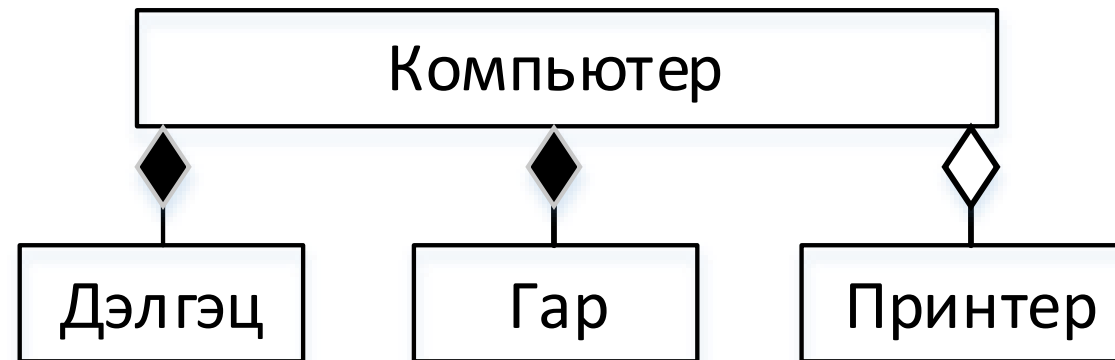
Бүрдмэл харьцааны хэлбэр





Бүтэцлэг / Composition

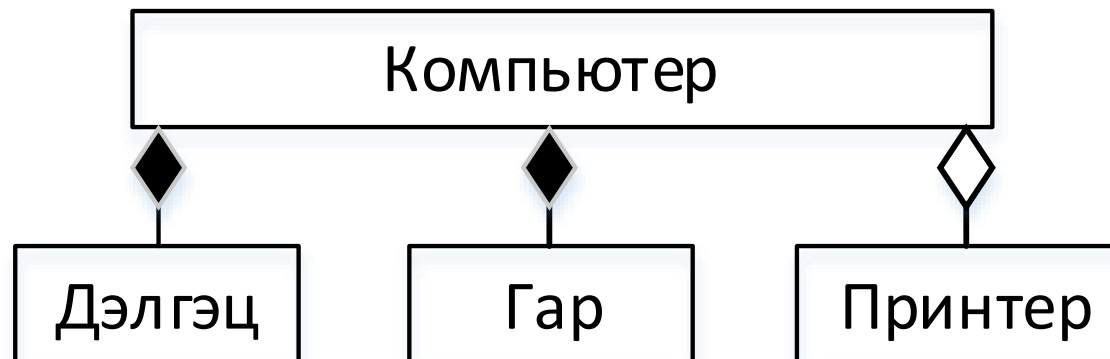
- Дэлгэц, гар нь компьютергүйгээр ажиллаж чадахгүй, хараат байдаг.
- Дангаар эзэмших тул “has a” холбоос байна.





Нийлмэл / Aggregation

- Принтер нь компьютергүйгээр ажиллаж чадна, бие даасан. Сүлжээнд залгаад бас ажиллаж чадна.
- Дундаа хэрэглэх үед “uses” холбоос байна.
- Түр эзэмших үед “has a” холбоос байна.





Has a

Uses



```
public class Pilot {
    private String name;
    public Pilot(String name) {
        this.name = name;
    }
    public String toString() {
        return "name=" + name;
    }
}
```

Engine

-speed: double
-type: String
+toString ()



Airplane

-model: String
-capacity: int
-engine: Engine
-pilot: Pilot
+assignPilot ()
+toString()



Pilot

-name: String
+toString()



```
public class Airplane {
    private String model;
    private int capacity;
    private Engine engine;
    private Pilot pilot;

    public Airplane(String model, int capacity, Engine engine, Pilot pilot) {
        this.model = model;
        this.capacity = capacity;
        this.engine = engine;
        this.pilot = pilot;
    }
}
```

```
public class Engine {
    private double speed;
    private String type;

    public Engine(double speed, String type) {
        this.speed = speed;
        this.type = type;
    }

    public String toString() {
        return "speed=" + speed + ", type=" + type;
    }
}
```

```
public class MainProgramm {
    public static void main(String args[]) {
        Pilot man1 = new Pilot("Bold");
        Pilot man2 = new Pilot("Bayar");
        Airplane airplane = new Airplane("Boeing 747", 350);
        airplane.assignPilot(man1);
        airplane.assignPilot(man2);
        System.out.println(airplane);
    }
}
```

Бүрдмэл харьцааны хэрэглэгдэх хэлбэрүүд

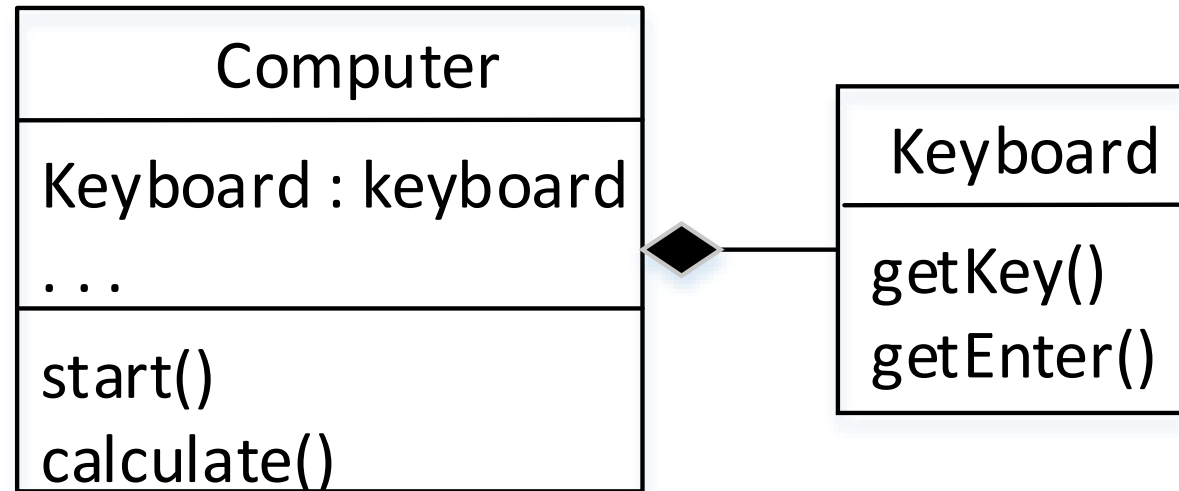


	Бүтэцлэг has a	Нийлмэл has a	Association uses
1. Гишүүн өгөгдөл хэлбэрээр,	X	X	
2. Параметр хэлбэрээр,	X	X	X
3. Буцаах төрөл хэлбэрээр,	X	X	
4. Үйлдэл дотор зарлах хэлбэрээр.			X



Гишүүн өгөгдөл хэлбэрээр хэрэглэх - Бүтэцлэг

```
class Computer {  
    private Keyboard keyboard;  
  
    ...  
}  
...  
class Keyboard {  
    ...  
}
```





Параметр хэлбэрээр хэрэглэх - Бүтэцлэг

```
class Computer {  
    private Keyboard keyboard;  
  
    ...  
    public void setKeyboard(Keyboard kb) {  
        keyboard = kb;  
    }  
}
```

Буцаах төрөл хэлбэрээр хэрэглэх - Бүтэцлэг



```
class Computer {  
    private Keyboard keyboard;  
  
    ...  
    public Keyboard getKeyboard() {  
        return keyboard;  
    }  
}
```

Нийлмэл (has a)



```
class Computer {  
    private Printer printer;  
    public void connectPrinter(Printer pr) {  
        printer = pr;  
    }  
    public Printer getCurrentPrinter()  
    { return printer; }  
}  
...
```

Нийлмэл (uses)



```
class Computer { ...  
    public printDocument(Printer pr) { ...  
    }  
}  
  
...  
class Printer {  
    ...  
}
```


Нийлмэл (uses)



```
class Computer { ...  
    public printDocument() {  
        Printer pr = new Printer();  
        ...  
    }  
}
```

Бүрдмэл харьцаа – объектуудын харилцан үйлчлэл

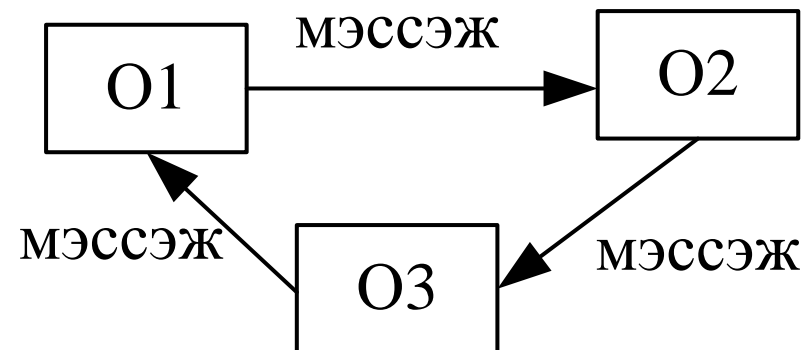


- “ОХ програм нь объектуудын харилцан үйлчлэлийн нэгдэл юм. Объект бүр тодорхой үүрэг гүйцэтгэнэ. Объект бүр бусад объектуудад хэрэгтэй үйлдэл эсвэл үйлчилгээгээр хангана” гэж үзсэн.
- Энэ харилцан үйлчлэлийг бүрдмэл харьцааны тусламжтай гүйцэтгэнэ. Бүрдмэл харьцаа нь хоорондоо холбогдох боломжийг бүрдүүлнэ. Классын үйлдэл нь харилцан үйлчлэл болно.



Объектуудын харилцан үйлчлэл

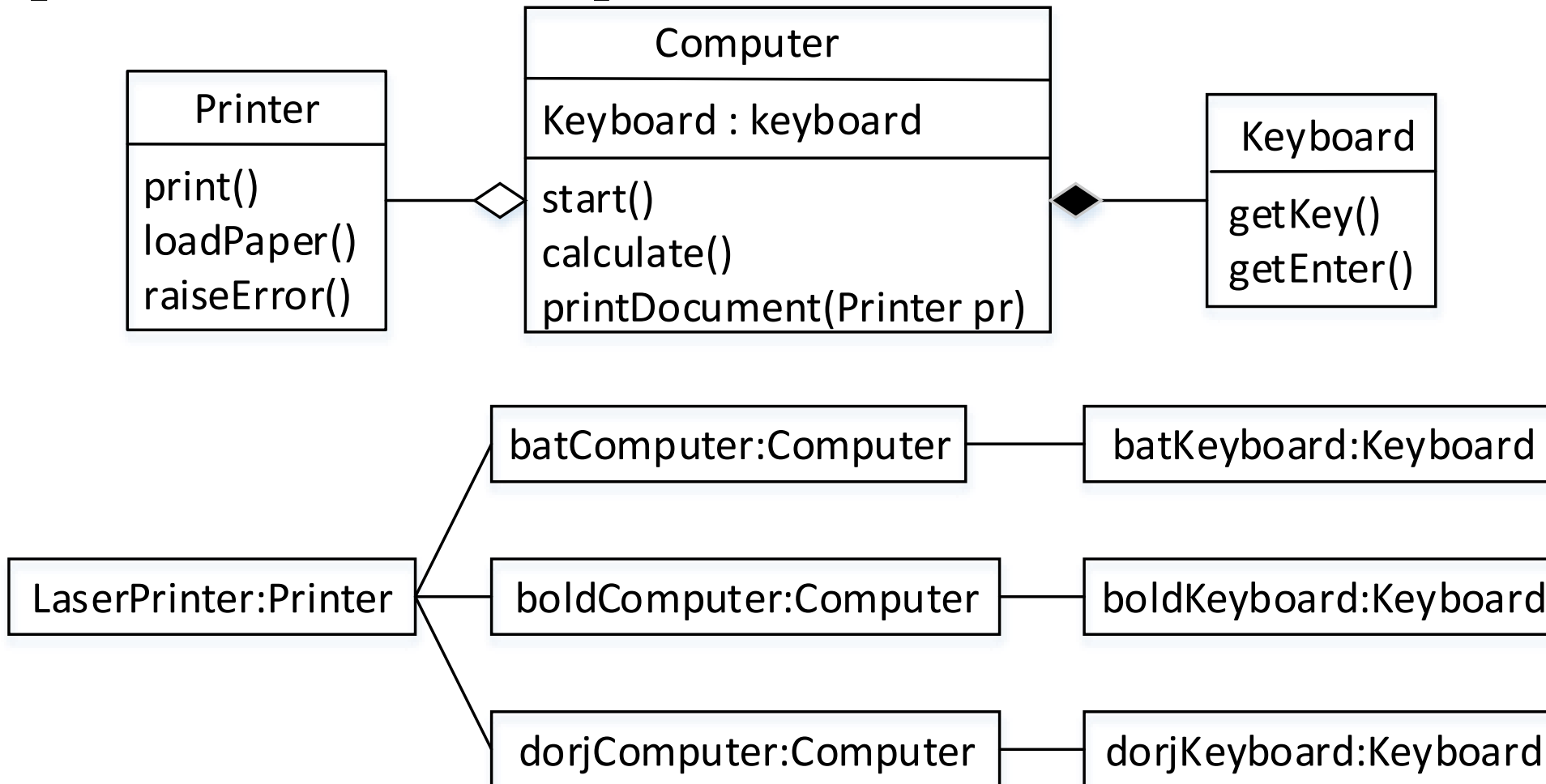
- Объектуудын харилцан үйлчлэл мэссэж дамжуулалтаар (үйлдэл дуудалтаар) гүйцэтгэгдэнэ.
- Мэссэж нь дуудсан объектын үйлдлийг идэвхжүүлнэ.
- O1 объект O2 дээрх нэг үйлдлийг дуудах замаар O2 объекттой харилцан үйлчлэлцэнэ. Тэр үйлдэл нь public байна. Үүнийг O1 O2 лүү мэссэж илгээх гэнэ.
- O1 болон O2 хоорондоо холбоотой байх ёстой. Энэ нь бүрдмэл харьцаагаар хэрэгжинэ.
- Үйлдэл дуудах нь C эсвэл Pascal зэрэг програмчлалын хэлний функц, процедур дуудахтай ижил ойлголт юм.





Классын бүрдмэл харьцаа

- Классын бүрдмэл харьцааны цаана объектуудын харилцан үйлчлэл байдаг. Учир нь:



Жишээ



The image displays a UML class diagram and two corresponding Java code snippets. The UML diagram shows three classes: **Computer**, **Printer**, and **Program**. **Computer** has a dashed arrow pointing to **Printer** and a dashed arrow pointing to **Program**. **Program** has a dashed arrow pointing to **Computer**.

The first code snippet is for the **Computer** class:

```
1 public class Computer {  
2     private Printer printer;  
3     public Printer getPrinter() {  
4         return printer;  
5     }  
6 }
```

The second code snippet is for the **Program** class:

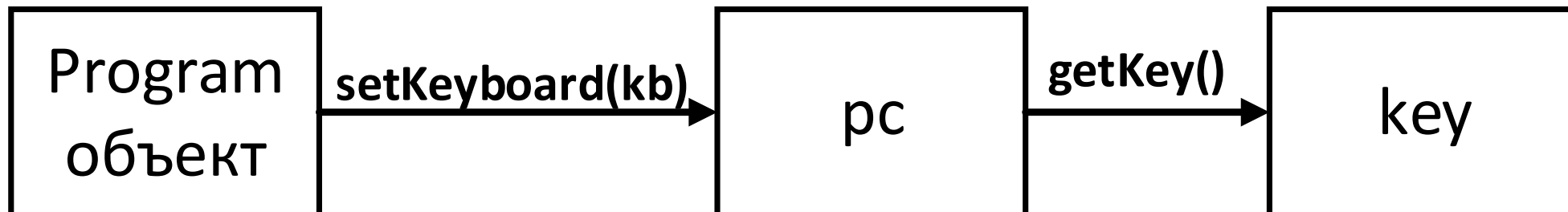
```
1 public class Program {  
2     public static void main() {  
3         Computer pc = new Computer();  
4     }  
5 }
```

Мэссэж дамжуулалт буюу функц дуудалт






```
class Program {  
    public static void main() {  
        Computer pc = new Computer();  
        Keyboard kb = new Keyboard();  
        pc.setKeyboard(kb);  
        ...  
    }  
}
```

```
class Computer {  
    Keyboard key;  
    public void setKeyboard(Keyboard k)  
    { key = k; }  
    public String typing() {  
        ...  
        char ch = key.getKey();  
        ...  
    }  
}
```

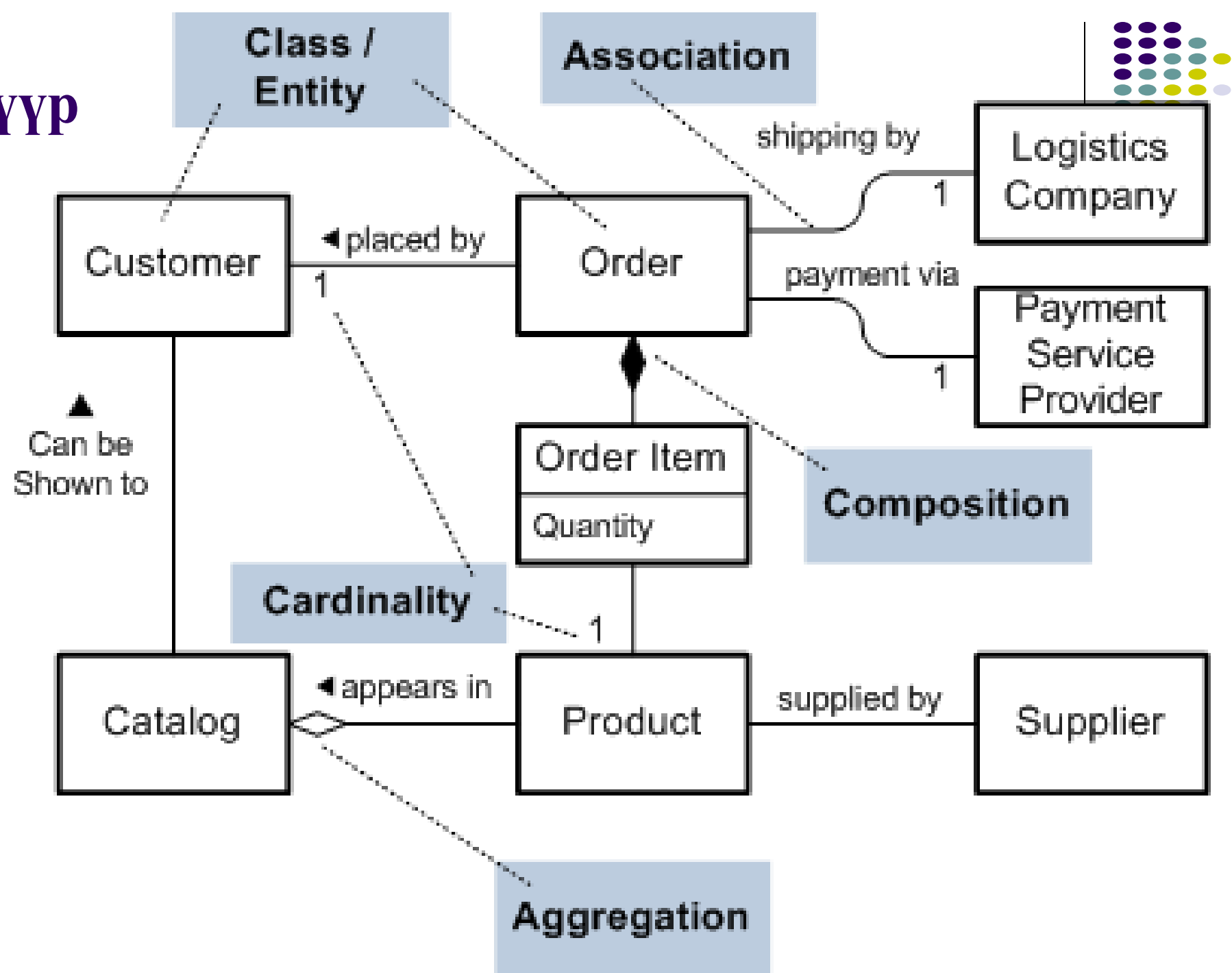


Association – Aggregation - Composition



	Association	Aggregation	Composition
Эзэмшигч	Эзэмшигч байхгүй	Нэг эзэмшигчтэй	Нэг эзэмшигчтэй
Амьдралын хугацаа	Өөрийн гэсэн амьдралын хугацаатай	Өөрийн гэсэн амьдралын хугацаатай	Эзэмшигчийн амьдралын хугацаагаар
Хүү объект	Ямар ч хүү объектгүй бүгд биеэ даасан	Хүү объект нь нэг л эцэгт харьяалагдана	Хүү объект нь нэг л эцэгт харьяалагдана
Холбоосын хүч		Сул	Хүчтэй
Тэмдэглэгээ			
Холбоос	uses	has a	owns, part of
Жишээ		Team - Player School - Student	Player - Award School - Department

Онлайн дэлгүүр



ДҮГНЭЛТ



- Бүрдмэл харьцаа нь объектуудын хоорондын харилцан үйлчлэлээр илэрнэ.
- Нэг объект нөгөө объектынхоо нэг хэсэг нь болж байгаа эсэхээр тодорхойлогдоно. Гэхдээ логик ойлголт.
- Aggregation ба Composition нь Association-ий онцлог хэлбэрүүд бөгөөд харилцан үйлчлэлийн хүчтэй сулаараа ялгаатай.
- Бүрдмэл харьцааг ЮМЛ диаграмаар хэрхэн дүрслэхийг үзлээ.
- Бүрдмэл харьцааны хэлбэрүүдийг хэрэгжүүлэх аргуудтай танилцлаа.