

Lesson 17

Polymorphism(Polimorfizm)

Encapsulation(Inkapsulyatsiya)

Packages(Paketlar)

Constructor(Konstruktorlar)



Polymorphism(Polimorfizm)

Polimorfizm(yunoncha “Ko`p shakllik” “Ko`p formalik” degan ma`nolarni anglatadi.)-bu bir xil harakatni har xil bajara olishimizdir.Masalan rul mexanizmini olaylik,rul yengilda kichikroq yuk mashinasida kattaroq yoki avtomashinalarda doira shaklida moto va velolarda to`g`ri bo`ladi lekin barchasi bir vazifa yani boshqaruv uchun ishlatiladi.

Polimorfizm ikkiga bo`linadi:

1)Overload

2)Override

Polymorphism(Polimorfizm) - Overload

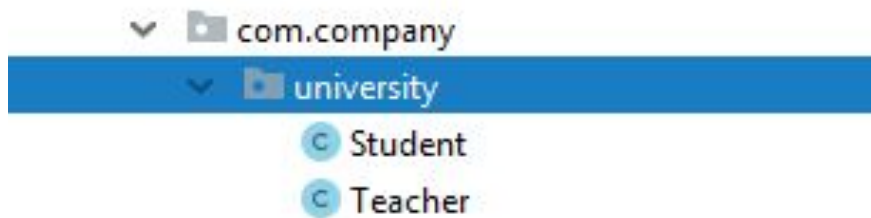


```
public int Square(int a,int b){  
    return a*b;  
}
```

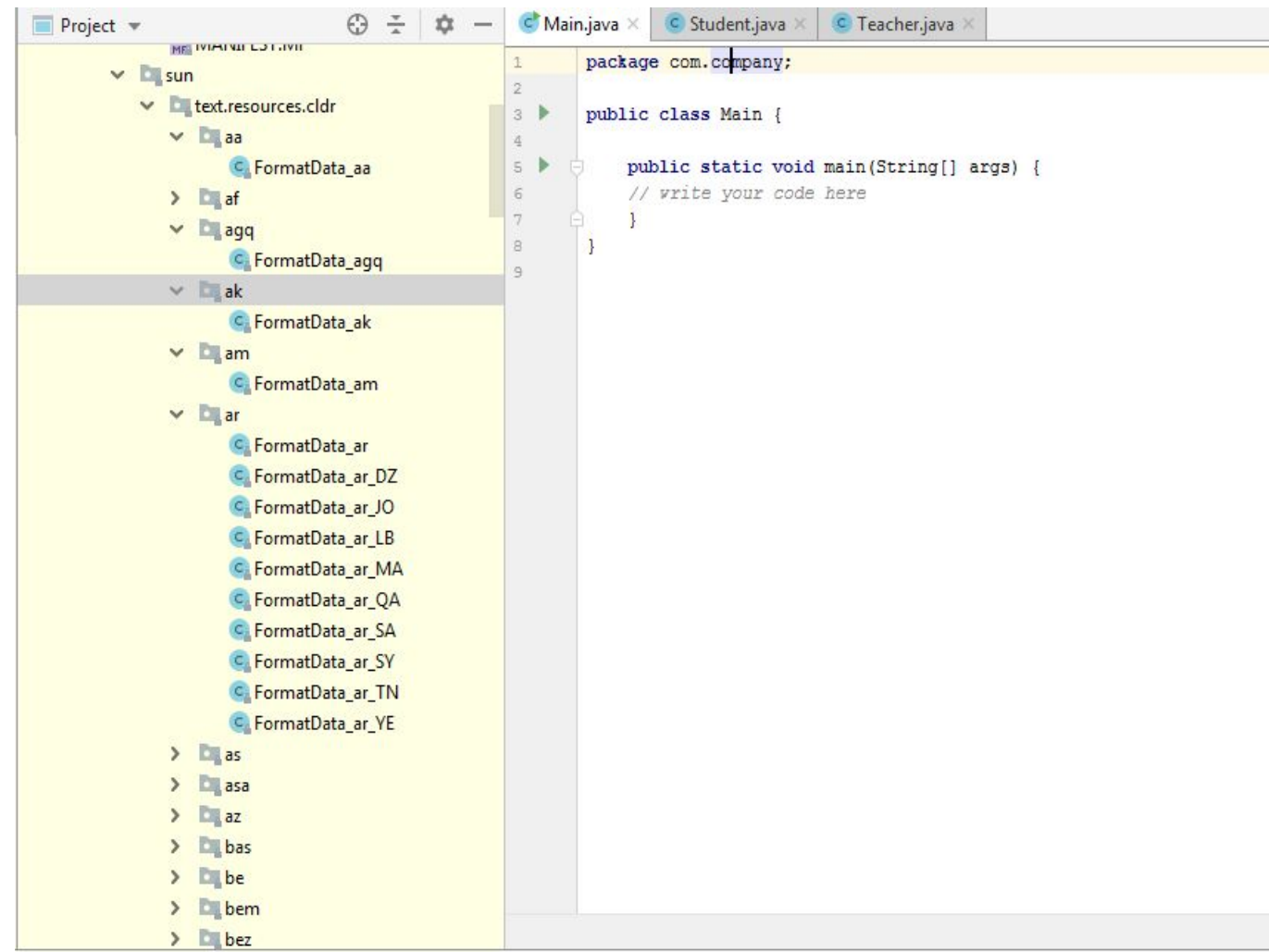
```
public double Square(int a,int b,int c){  
    double p,s;  
    p=(a+b+c)/2;  
    s=Math.sqrt(p*(p-a)*(p-b)*(p-c));  
    return s;  
}
```

```
public int Square(int n){  
    return n*n;  
}
```

Packages(Paketlar)



Paket – bu bir toifa yoki bir tushinchaga oid classlar va bosh fayllar saqlanadigan papka hisoblanadi. Va biz klasslarga paketlar orqali murojat qilamiz va paketlarning shrafoti bilan biz turli paketlarda birxil nomli fayllarni saqlay olamiz: `package com.company.university;`





Encapsulation(Inkapsulyatsiya)

Inkapsulatsiya – Javada classlar maydoni va metodlarini notog`ri ishlatish ya`ni to`g`ridan to`g`ri emas qanaqadir boshqa cheklangan yo`llar orqali ishlatishga aytiladi. Inkapsulyatsiya- kapsula ichida ma`nosini bildirib classlar maydon va metodlari 4 xil ximoya bilan ximoyalanadi.

Bular:

- **Public - barcha joy(paket class voris klass)dan murojat qilish mumkin.**
- **Protected – bir paketning ichida murojat qilish va voris klassdan murojat qilish mumkin**
- **Default – bir paketda murojat qilish mumkin**
- **Private – faqat classning ichidan turib murojat qilish mumkin**

Encapsulation(Inkapsulyatsiya)



| Modifier | Class | Package | Subclass | Other Classes |
|-------------|-------|---------|----------|---------------|
| Private | Yes | No | No | No |
| No modifier | Yes | Yes | No | No |
| Protected | Yes | Yes | Yes | No |
| Public | Yes | Yes | Yes | Yes |

Encapsulation(Inkapsulyatsiya)



```
public class Student {  
  
    public String name;  
  
    protected String surname;  
  
    int age;  
  
    private double money;  
  
}
```



Constructor(Konstruktor)

Konstruktor – javada klassning namunasi(obyekti)ni yaratishda ishlatiladi va istalgan classda hech bo`lmaganda bitta konstruktor bo`ladi.

```
class ClassName{  
    public ClassName(){  
  
    }  
}
```

Agar siz bitta bo`lsa ham konstruktor yaratmasangiz default konstruktor yaratiladi.



Destructor(Destruktor)

Destruktor – javada klassning namunasi(obyekti)ni yaratilganda u xotiradan joy oladi va u joyni object ishlatilib bo`lingandan so`ng o`chirishga to`g`ri keladi va java da bu ish finalize()metodi orqali amalga oshiriladi. Javada

```
class ClassName{
```

- **@Override**
 - **protected void finalize() throws Throwable {**
 - **super.finalize();**
 - **}**
- ```
}
```



# Class – constructor and destructor

---

```
class Kasr{
 Kasr(){
 //...
 }

 Kasr(int a){
 //...
 }
}
```

```
class Kasr{

 @Override
 protected void finalize() throws Throwable {
 super.finalize();
 }
}
```

# Konstruktor(Konstruktor)

---



```
package com.company.university;
```

```
public class Student {
 private String name;
 private String surname;
 private int age;
 private double money;
 public Student() {

 }
 public Student(String name, String surname, int age) {
 this.name = name;
 this.surname = surname;
 this.age = age;
 }
 public Student(String name, String surname, int age, double money) {
 this.name = name;
 this.surname = surname;
 this.age = age;
 this.money = money;
 }
}
```



# Static(Statik)

---

Static qotib qolgan ma`nosini berib static kalit so`zi maydon va me`todlar oldidan ishlatiladi. Static maydon – class obyektini emas balki uning ozi(class)ning nomidan ishga tushadigan va doim xotirani egallab turadigan qismi hisoblanadi.

Static metod ham – class obyekti nomidan emas balki o`zining nomidan ishga tushadi.

```
public class Student {
 private static double money;

 public Student() {
 }

 public static double getMoney() {
 return money;
 }

 public static void setMoney(double money) {
 Student.money = money;
 }
}
```

The end