

Week 7 Homework Q26

Telmen Enkhbold

San Fransico Bay University

CE480 - Java and Internet Application

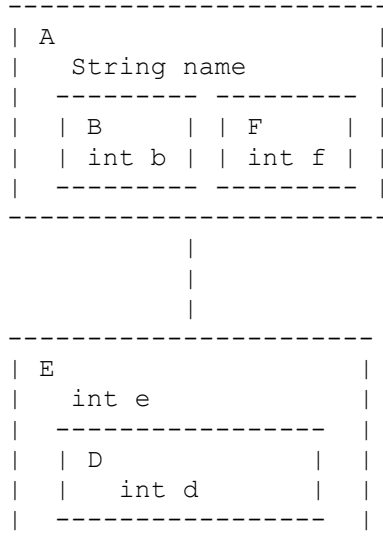
Dr. Chang, Henry

10/12/2023

Author Note

The Question

1. Inheritance + Aggregation



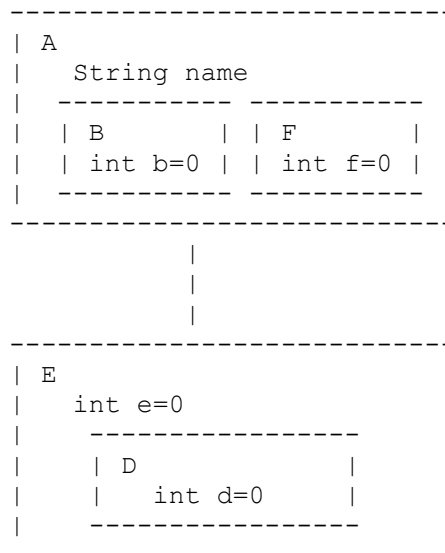
Note:

- Each class must implement 4 types of member functions
 - Helping function
 -
 - ```
private void trace(String s) {
 System.out.println(s);
}
```
  - Access functions
    - get
    - set
    - predicate
      - class A, B, D, E, F
      - 
      - `isLargeValue()`

==> Check whether the values of all its attributes are greater than 100. For example, class E's isLargeValue() returns true if all the values of b, f, d, e are greater than 100.

- `class A`
  - 
  - `isBruceLee()`
    - `==>` Check whether the value of the attribute "name" is "Bruce Lee".
- Implementor function
  - `class A, B, D, E, F`
  - 
  - `changetoZero()`

==> The values of all its numerical attributes are set to 0. For example, class E's `changeloZero()` has this result:



class A

```
changeName(String new name)
```

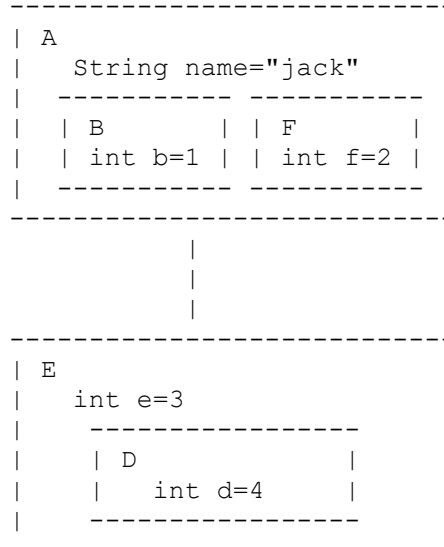
==> The value of the attribute "name" is changed to a new name.

class A, B, D, E, F

- toString()
- clone()
- equals()

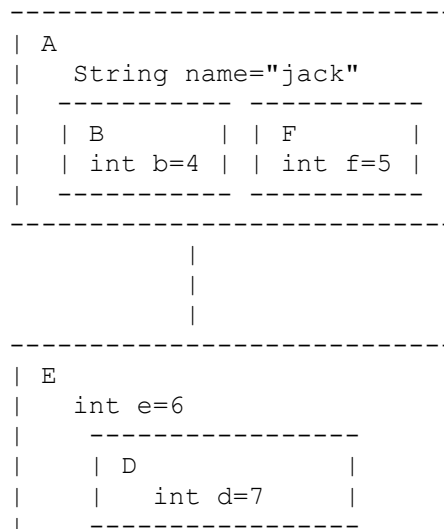
main function

Create this object eObj



Display the values of all the attributes of eObj

Change the values of eObj to



-----

Display the values of all the attributes of eObj

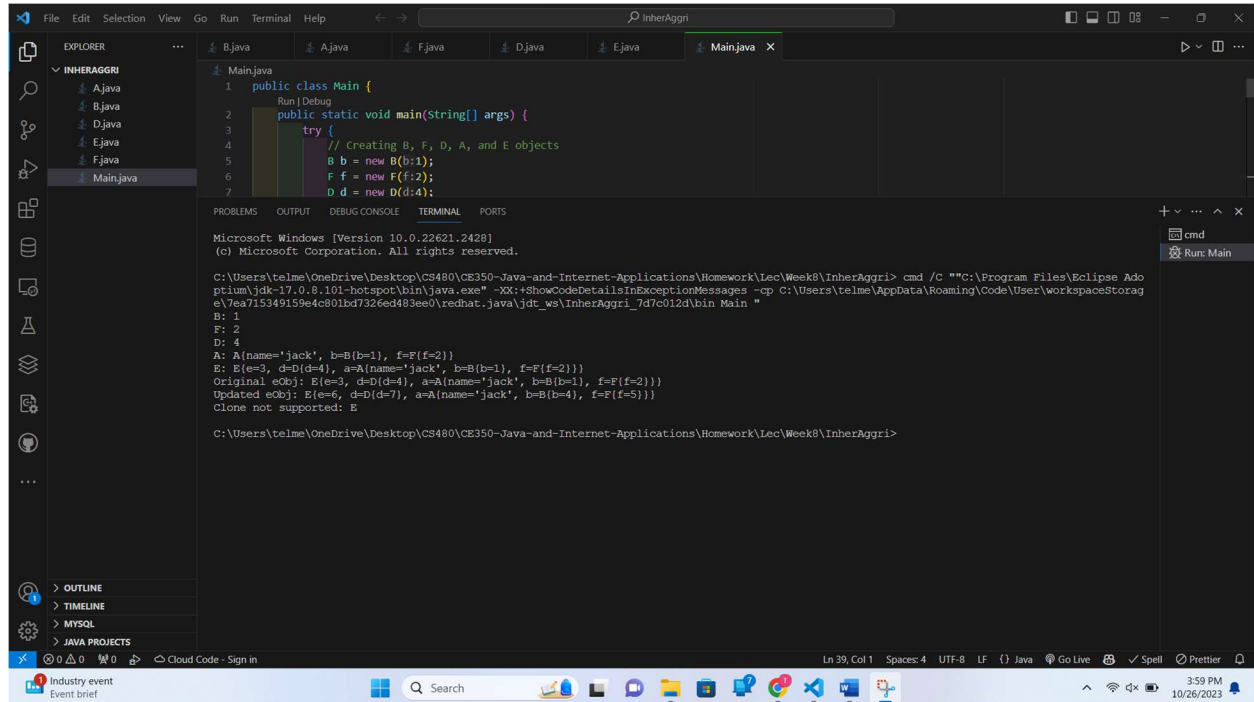
Clone the eObj to create eObj1

Display the values of all the attributes of eObj1

Check whether eObj is equal to eObj1

- References
  - [Class containing two layers of simple object attribute](#)
  - [One layer of simple object attribute + toString\(\) + clone\(\) + equals\(\)](#)
  - [get/set](#)
  - [4 Types of Member Functions](#)
  - [Summary of toString\(\), clone\(\), equals\(\)](#)
  - [Inheritance Class Structure](#)

## Screenshot



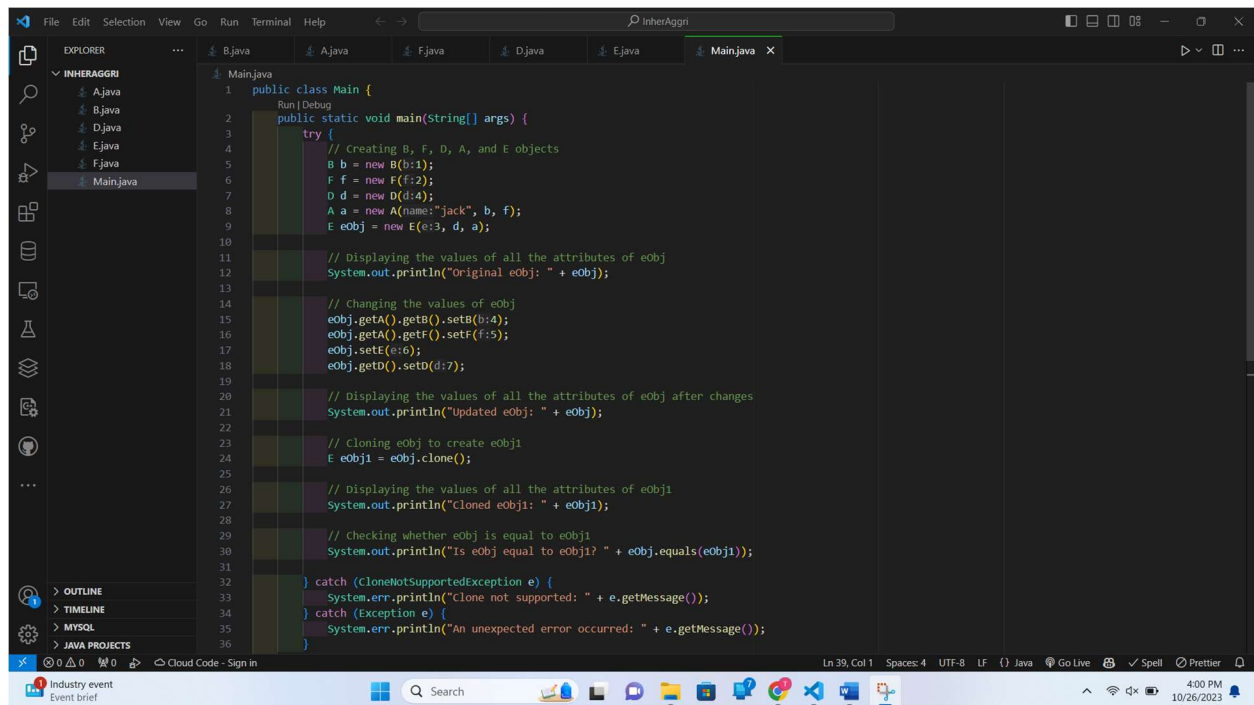
```
1 public class Main {
2 public static void main(String[] args) {
3 try {
4 // Creating B, F, D, A, and E objects
5 B b = new B(b:1);
6 F f = new F(f:2);
7 D d = new D(d:4);
8 A a = new A(name="jack", b=B(b=1), f=F(f=2));
9 E eObj = new E(e=3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)));
10 System.out.println("original eObj: " + eObj);
11 // Changing the values of eObj
12 eObj.getA().getB().setB(b:4);
13 eObj.getA().getF().setF(f:5);
14 eObj.setE(e:6);
15 eObj.getD().setD(d:7);
16 // Displaying the values of all the attributes of eObj after changes
17 System.out.println("Updated eObj: " + eObj);
18 // Cloning eObj to create eObj1
19 E eObj1 = eObj.clone();
20 // Displaying the values of all the attributes of eObj1
21 System.out.println("Cloned eObj1: " + eObj1);
22 // Checking whether eObj is equal to eObj1
23 System.out.println("Is eObj equal to eObj1? " + eObj.equals(eObj1));
24 } catch (CloneNotSupportedException e) {
25 System.err.println("Clone not supported: " + e.getMessage());
26 } catch (Exception e) {
27 System.err.println("An unexpected error occurred: " + e.getMessage());
28 }
29 }
30 }
```

Microsoft Windows [Version 10.0.22621.2428]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\telme\OneDrive\Desktop\CS480\CE350-Java-and-Internet-Applications\Homework\Lec\Week8\InherAggri> cmd /C ""C:\Program Files\Eclipse\Adoptium\jdk-17.0.8-hotspot\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\telme\AppData\Roaming\Code\User\workspaceStorage\ea7ea715349159e4c801bd7326ed483ee0\redhat.java\jdt\_ws\InherAggri\_\d7c012d\bin Main "

B: 1  
F: 2  
D: 4  
A: A(name="jack", b=B(b=1), f=F(f=2))  
E: E(e=3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)))  
Original eObj: E(e=3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)))  
Updated eObj: E(e=6, d=D(d=7), a=A(name="jack", b=B(b=4), f=F(f=5)))  
Clone not supported: E

C:\Users\telme\OneDrive\Desktop\CS480\CE350-Java-and-Internet-Applications\Homework\Lec\Week8\InherAggri>



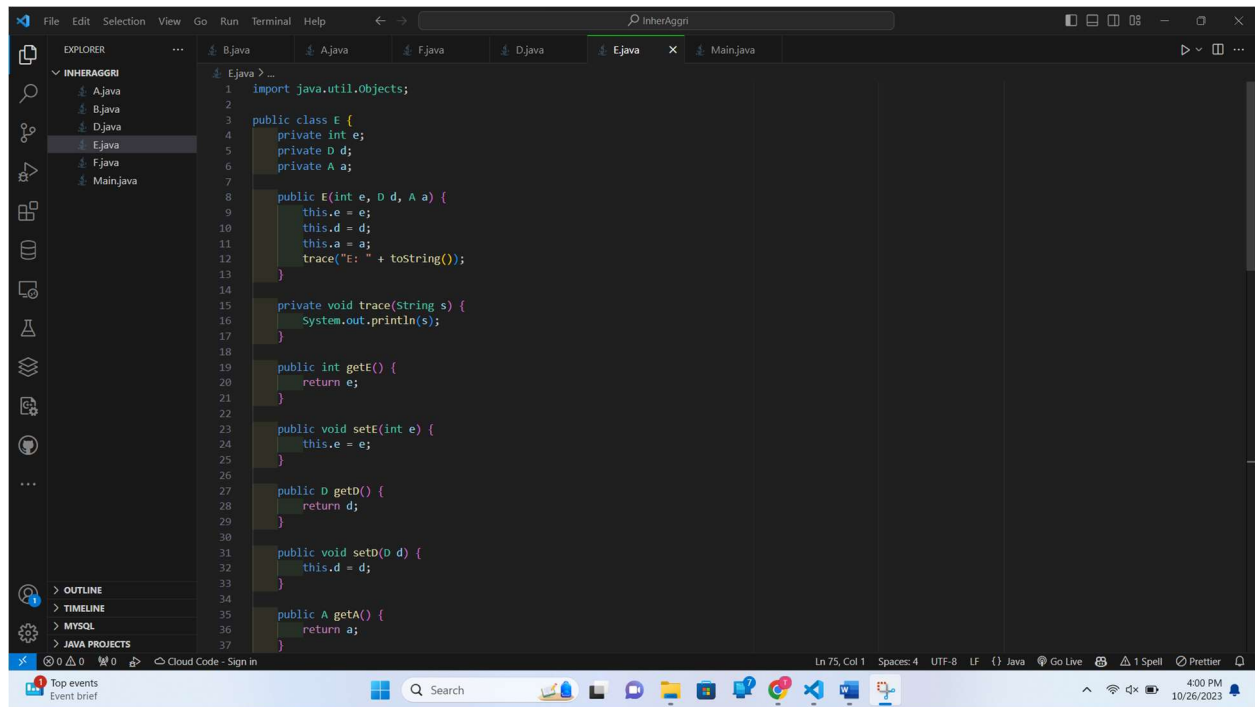
```
1 public class Main {
2 public static void main(String[] args) {
3 try {
4 // Creating B, F, D, A, and E objects
5 B b = new B(b:1);
6 F f = new F(f:2);
7 D d = new D(d:4);
8 A a = new A(name="jack", b=B(b=1), f=F(f=2));
9 E eObj = new E(e:3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)));
10 System.out.println("original eObj: " + eObj);
11 // Changing the values of eObj
12 eObj.getA().getB().setB(b:4);
13 eObj.getA().getF().setF(f:5);
14 eObj.setE(e:6);
15 eObj.getD().setD(d:7);
16 // Displaying the values of all the attributes of eObj after changes
17 System.out.println("Updated eObj: " + eObj);
18 // Cloning eObj to create eObj1
19 E eObj1 = eObj.clone();
20 // Displaying the values of all the attributes of eObj1
21 System.out.println("Cloned eObj1: " + eObj1);
22 // Checking whether eObj is equal to eObj1
23 System.out.println("Is eObj equal to eObj1? " + eObj.equals(eObj1));
24 } catch (CloneNotSupportedException e) {
25 System.err.println("Clone not supported: " + e.getMessage());
26 } catch (Exception e) {
27 System.err.println("An unexpected error occurred: " + e.getMessage());
28 }
29 }
30 }
```

Microsoft Windows [Version 10.0.22621.2428]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\telme\OneDrive\Desktop\CS480\CE350-Java-and-Internet-Applications\Homework\Lec\Week8\InherAggri> cmd /C ""C:\Program Files\Eclipse\Adoptium\jdk-17.0.8-hotspot\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\telme\AppData\Roaming\Code\User\workspaceStorage\ea7ea715349159e4c801bd7326ed483ee0\redhat.java\jdt\_ws\InherAggri\_\d7c012d\bin Main "

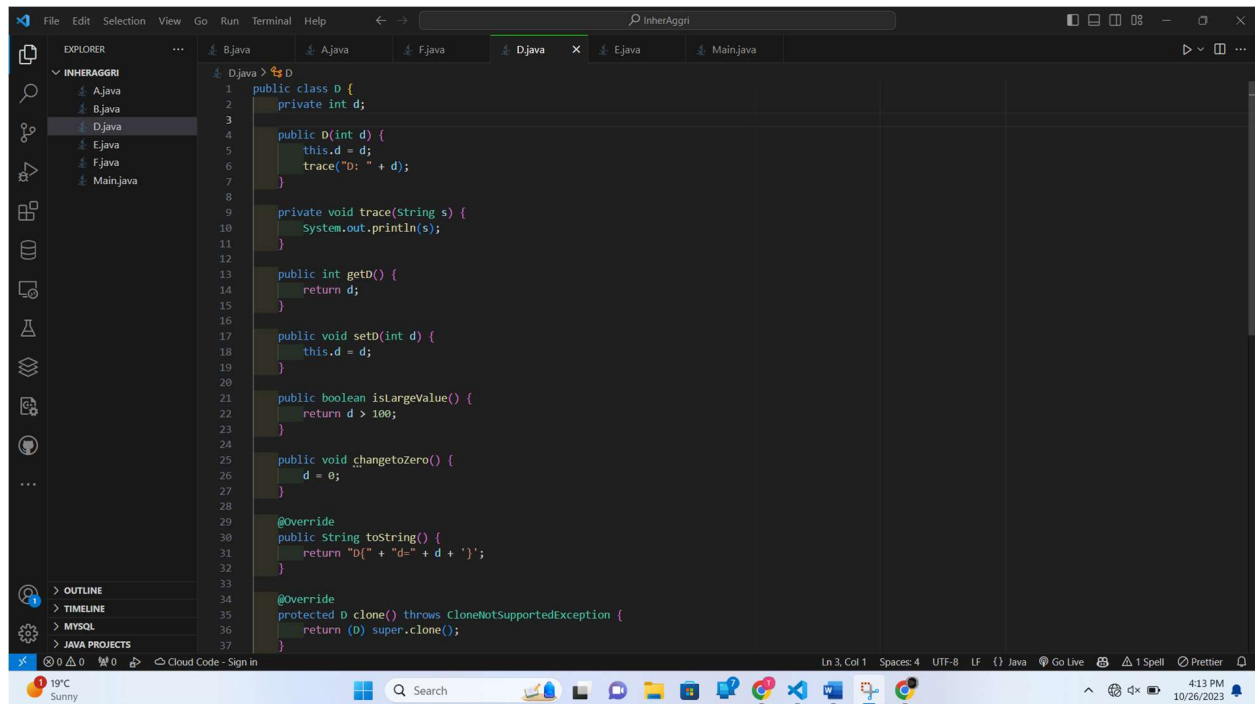
B: 1  
F: 2  
D: 4  
A: A(name="jack", b=B(b=1), f=F(f=2))  
E: E(e=3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)))  
Original eObj: E(e=3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)))  
Updated eObj: E(e=6, d=D(d=7), a=A(name="jack", b=B(b=4), f=F(f=5)))  
Cloned eObj1: E(e=3, d=D(d=4), a=A(name="jack", b=B(b=1), f=F(f=2)))  
Is eObj equal to eObj1? false  
Clone not supported: E

C:\Users\telme\OneDrive\Desktop\CS480\CE350-Java-and-Internet-Applications\Homework\Lec\Week8\InherAggri>



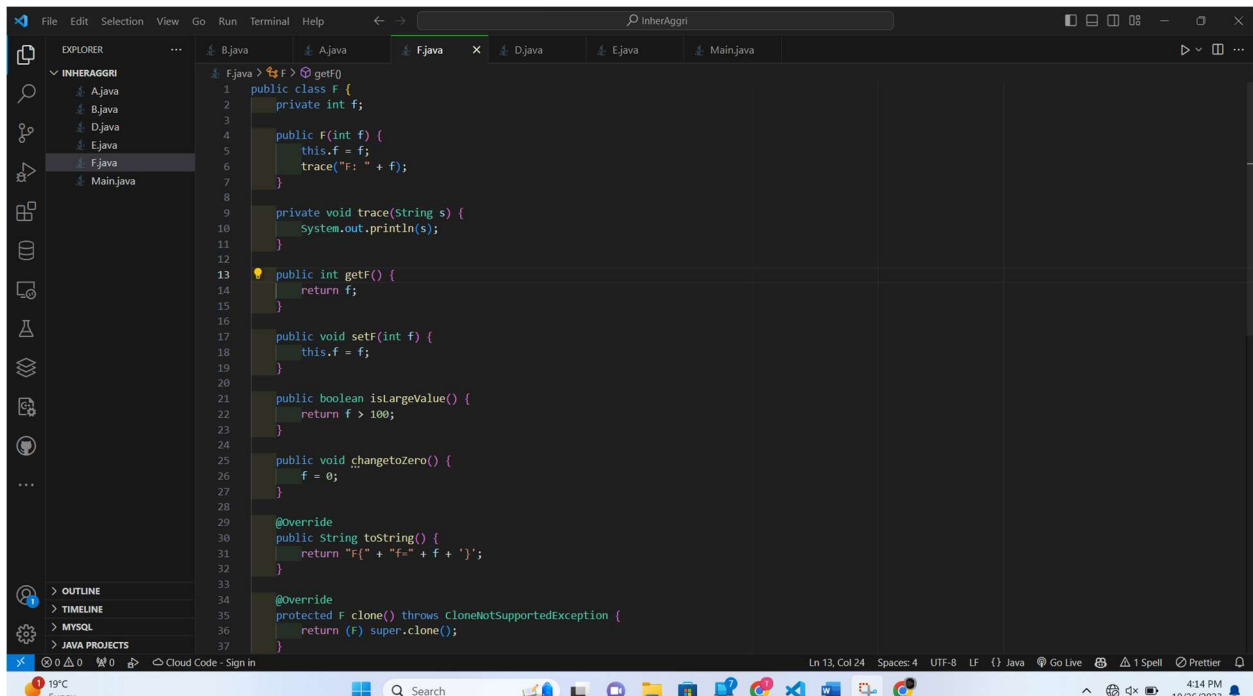
The screenshot shows the Visual Studio Code editor with the 'InherAggri' project open. The Explorer sidebar on the left shows the project structure with files: B.java, A.java, D.java, E.java, F.java, and Main.java. The E.java file is selected and its code is displayed in the editor. The code defines a class E that aggregates other classes (A, D) and has its own attributes and methods.

```
1 import java.util.Objects;
2
3 public class E {
4 private int e;
5 private D d;
6 private A a;
7
8 public E(int e, D d, A a) {
9 this.e = e;
10 this.d = d;
11 this.a = a;
12 trace("E: " + toString());
13 }
14
15 private void trace(String s) {
16 System.out.println(s);
17 }
18
19 public int getE() {
20 return e;
21 }
22
23 public void setE(int e) {
24 this.e = e;
25 }
26
27 public D getD() {
28 return d;
29 }
30
31 public void setD(D d) {
32 this.d = d;
33 }
34
35 public A getA() {
36 return a;
37 }
```



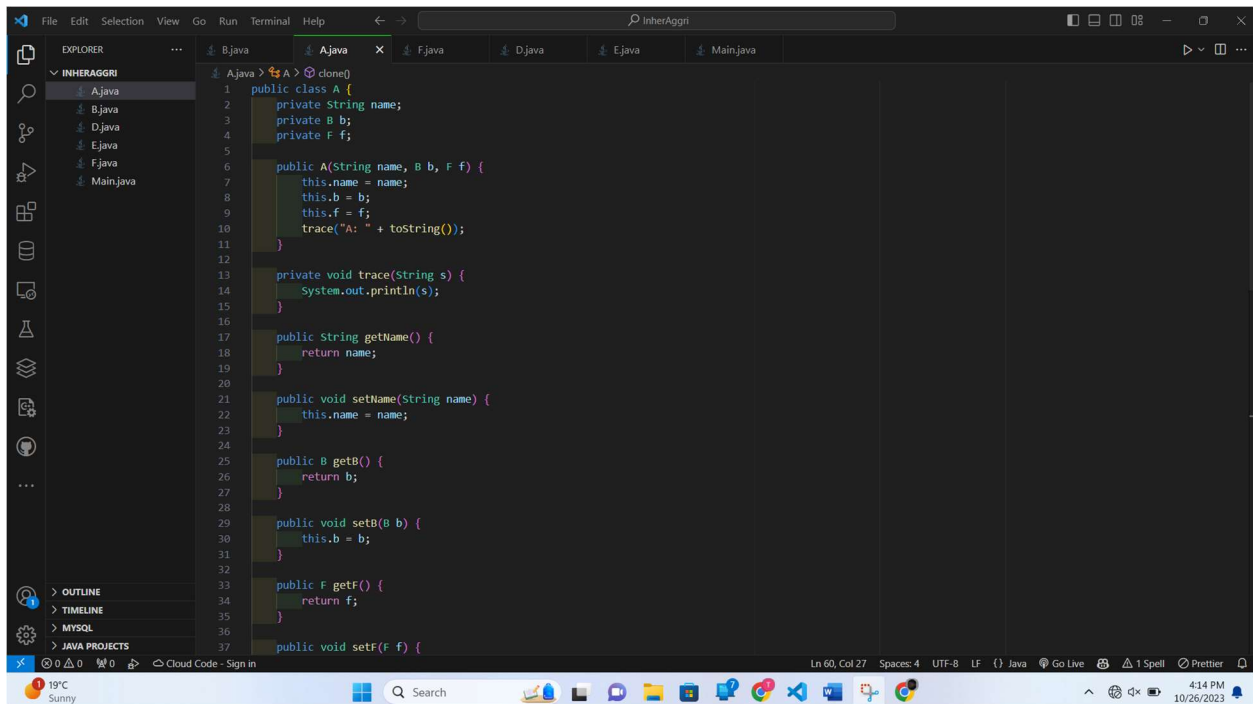
The screenshot shows the Visual Studio Code editor with the 'InherAggri' project open. The Explorer sidebar on the left shows the project structure with files: B.java, A.java, D.java, E.java, F.java, and Main.java. The D.java file is selected and its code is displayed in the editor. The code defines a class D with attributes, methods, and overrides.

```
1 public class D {
2 private int d;
3
4 public D(int d) {
5 this.d = d;
6 trace("D: " + d);
7 }
8
9 private void trace(String s) {
10 System.out.println(s);
11 }
12
13 public int getD() {
14 return d;
15 }
16
17 public void setD(int d) {
18 this.d = d;
19 }
20
21 public boolean isLargeValue() {
22 return d > 100;
23 }
24
25 public void changeToZero() {
26 d = 0;
27 }
28
29 @Override
30 public String toString() {
31 return "D{" + "d=" + d + '}';
32 }
33
34 @Override
35 protected D clone() throws CloneNotSupportedException {
36 return (D) super.clone();
37 }
```



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The Explorer sidebar shows a project named 'INHERAGGRI' with files 'A.java', 'B.java', 'D.java', 'E.java', 'F.java', and 'Main.java'. The 'F.java' file is selected and its code is displayed in the editor. The code defines a class 'F' with a private integer field 'f'. It includes a constructor 'F(int f)' that initializes 'f' and prints it. There are methods for 'getF()', 'setF(int f)', 'isLargeValue()', 'changeToZero()', 'toString()', and 'clone()'. The 'clone()' method is marked as 'protected' and throws 'CloneNotSupportedException'.

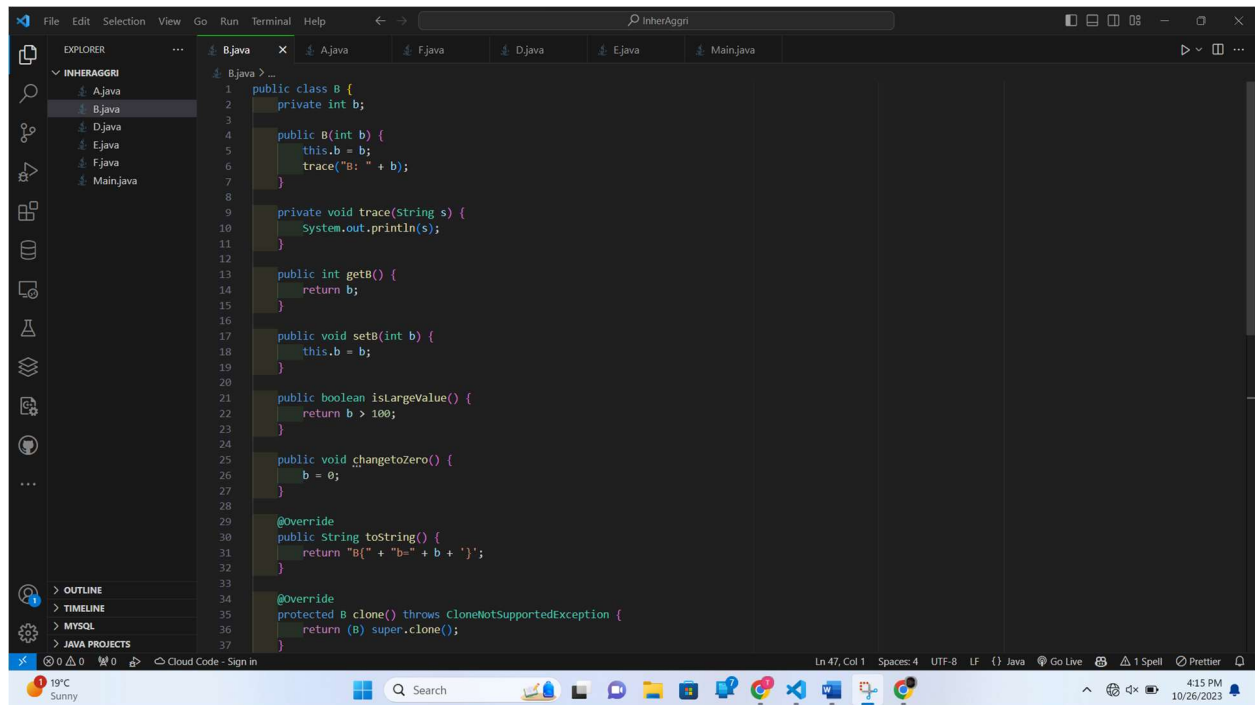
```
1 public class F {
2 private int f;
3
4 public F(int f) {
5 this.f = f;
6 trace("f: " + f);
7 }
8
9 private void trace(String s) {
10 System.out.println(s);
11 }
12
13 public int getF() {
14 return f;
15 }
16
17 public void setF(int f) {
18 this.f = f;
19 }
20
21 public boolean isLargeValue() {
22 return f > 100;
23 }
24
25 public void changeToZero() {
26 f = 0;
27 }
28
29 @Override
30 public String toString() {
31 return "F{" + "f=" + f + '}';
32 }
33
34 @Override
35 protected F clone() throws CloneNotSupportedException {
36 return (F) super.clone();
37 }
38}
```



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The Explorer sidebar shows a project named 'INHERAGGRI' with files 'A.java', 'B.java', 'D.java', 'E.java', 'F.java', and 'Main.java'. The 'A.java' file is selected and its code is displayed in the editor. The code defines a class 'A' with private fields 'String name', 'B b', and 'F f'. It includes a constructor 'A(String name, B b, F f)' that initializes these fields and prints them. There are methods for 'getName()', 'setName(String name)', 'getB()', 'setB(B b)', 'getF()', and 'setF(F f)'. The 'clone()' method is marked as 'protected' and throws 'CloneNotSupportedException'.

```
1 public class A {
2 private String name;
3 private B b;
4 private F f;
5
6 public A(String name, B b, F f) {
7 this.name = name;
8 this.b = b;
9 this.f = f;
10 trace("A: " + toString());
11 }
12
13 private void trace(String s) {
14 System.out.println(s);
15 }
16
17 public String getName() {
18 return name;
19 }
20
21 public void setName(String name) {
22 this.name = name;
23 }
24
25 public B getB() {
26 return b;
27 }
28
29 public void setB(B b) {
30 this.b = b;
31 }
32
33 public F getF() {
34 return f;
35 }
36
37 public void setF(F f) {
38 }
39}
```





---

### The Code

---

#### A.java

```
public class A {
 private String name;
 private B b;
 private F f;

 public A(String name, B b, F f) {
 this.name = name;
 this.b = b;
 this.f = f;
 trace("A: " + toString());
 }

 private void trace(String s) {
 System.out.println(s);
 }

 public String getName() {
 return name;
 }
}
```

```
public void setName(String name) {
 this.name = name;
}

public B getB() {
 return b;
}

public void setB(B b) {
 this.b = b;
}

public F getF() {
 return f;
}

public void setF(F f) {
 this.f = f;
}

public boolean isBruceLee() {
 return "Bruce Lee".equals(name);
}

public void changeName(String newName) {
 this.name = newName;
}

public void changetoZero() {
 b.changetoZero();
 f.changetoZero();
}

@Override
public String toString() {
 return "A{" + "name='" + name + '\'' + ", b=" + b + ", f=" + f +
 '}';
}

@Override
protected A clone() throws CloneNotSupportedException {
 A cloned = (A) super.clone();
 cloned.b = this.b.clone();
 return cloned;
}
```

```
}
```

B.java

```
public class B {
 private int b;

 public B(int b) {
 this.b = b;
 trace("B: " + b);
 }

 private void trace(String s) {
 System.out.println(s);
 }

 public int getB() {
 return b;
 }

 public void setB(int b) {
 this.b = b;
 }

 public boolean isLargeValue() {
 return b > 100;
 }

 public void changetoZero() {
 b = 0;
 }

 @Override
 public String toString() {
 return "B{" + "b=" + b + '}';
 }

 @Override
 protected B clone() throws CloneNotSupportedException {
 return (B) super.clone();
 }

 @Override
 public boolean equals(Object obj) {
 if (this == obj) return true;
 if (obj == null || getClass() != obj.getClass()) return false;
 }
}
```

```
 B b1 = (B) obj;
 return b == b1.b;
 }
}
```

D.java

```
public class D {
 private int d;

 public D(int d) {
 this.d = d;
 trace("D: " + d);
 }

 private void trace(String s) {
 System.out.println(s);
 }

 public int getD() {
 return d;
 }

 public void setD(int d) {
 this.d = d;
 }

 public boolean isLargeValue() {
 return d > 100;
 }

 public void changetoZero() {
 d = 0;
 }

 @Override
 public String toString() {
 return "D{" + "d=" + d + '}';
 }

 @Override
 protected D clone() throws CloneNotSupportedException {
 return (D) super.clone();
 }
}
```

```
}

@Override
public boolean equals(Object obj) {
 if (this == obj) return true;
 if (obj == null || getClass() != obj.getClass()) return false;
 D d1 = (D) obj;
 return d == d1.d;
}
}
```

E.java

```
import java.util.Objects;

public class E {
 private int e;
 private D d;
 private A a;

 public E(int e, D d, A a) {
 this.e = e;
 this.d = d;
 this.a = a;
 trace("E: " + toString());
 }

 private void trace(String s) {
 System.out.println(s);
 }

 public int getE() {
 return e;
 }

 public void setE(int e) {
 this.e = e;
 }

 public D getD() {
 return d;
 }
}
```

```
public void setD(D d) {
 this.d = d;
}

public A getA() {
 return a;
}

public void setA(A a) {
 this.a = a;
}

public boolean isLargeValue() {
 return e > 100 && d.isLargeValue() && a.getB().isLargeValue() &&
a.getF().isLargeValue();
}

public void changetoZero() {
 e = 0;
 d.changetoZero();
 a.changetoZero();
}

@Override
public String toString() {
 return "E{" + "e=" + e + ", d=" + d + ", a=" + a + '}';
}

@Override
protected E clone() throws CloneNotSupportedException {
 E cloned = (E) super.clone();
 cloned.d = this.d.clone();
 cloned.a = this.a.clone();
 return cloned;
}

@Override
public boolean equals(Object obj) {
 if (this == obj) return true;
 if (obj == null || getClass() != obj.getClass()) return false;
 E e1 = (E) obj;
 return e == e1.e && Objects.equals(d, e1.d) && Objects.equals(a,
e1.a);
}
}
```

Main.java

```
public class Main {
 public static void main(String[] args) {
 try {
 // Creating B, F, D, A, and E objects
 B b = new B(1);
 F f = new F(2);
 D d = new D(4);
 A a = new A("jack", b, f);
 E eObj = new E(3, d, a);

 // Displaying the values of all the attributes of eObj
 System.out.println("Original eObj: " + eObj);

 // Changing the values of eObj
 eObj.getA().getB().setB(4);
 eObj.getA().getF().setF(5);
 eObj.setE(6);
 eObj.getD().setD(7);

 // Displaying the values of all the attributes of eObj after
changes
 System.out.println("Updated eObj: " + eObj);

 // Cloning eObj to create eObj1
 E eObj1 = eObj.clone();

 // Displaying the values of all the attributes of eObj1
 System.out.println("Cloned eObj1: " + eObj1);

 // Checking whether eObj is equal to eObj1
 System.out.println("Is eObj equal to eObj1? " +
eObj.equals(eObj1));

 } catch (CloneNotSupportedException e) {
 System.err.println("Clone not supported: " + e.getMessage());
 } catch (Exception e) {
 System.err.println("An unexpected error occurred: " +
e.getMessage());
 }
 }
 }
}
```





Refrence

<https://hc.labnet.sfbu.edu/~henry/sfbu/course/introjava/inheritance/slide/exercises4.html>  
Links to an external site.

Q26 ==> Aggregation and inheritance

Github-<https://github.com/Georgycas/CE350-Java-and-Internet-Applications/tree/main/Homework/Lec/Week8/InherAggri>