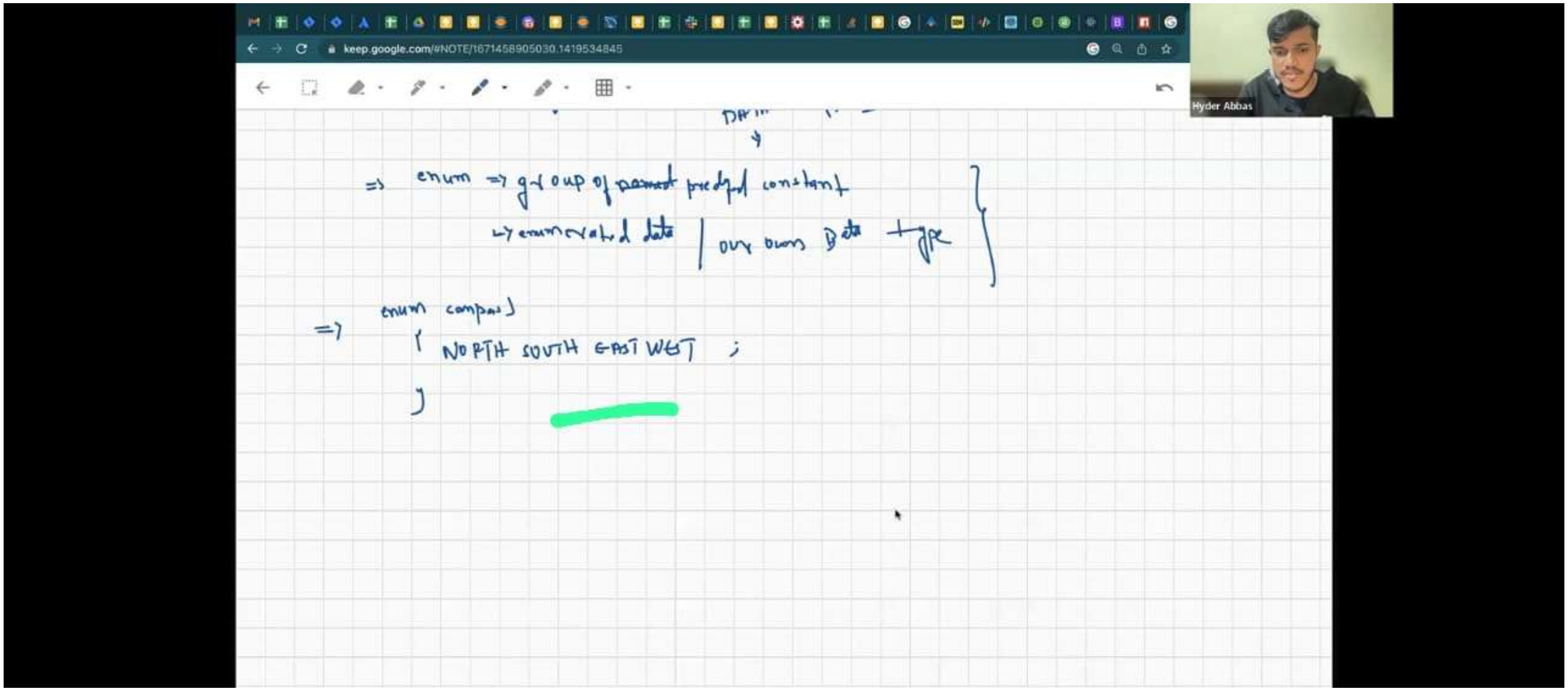
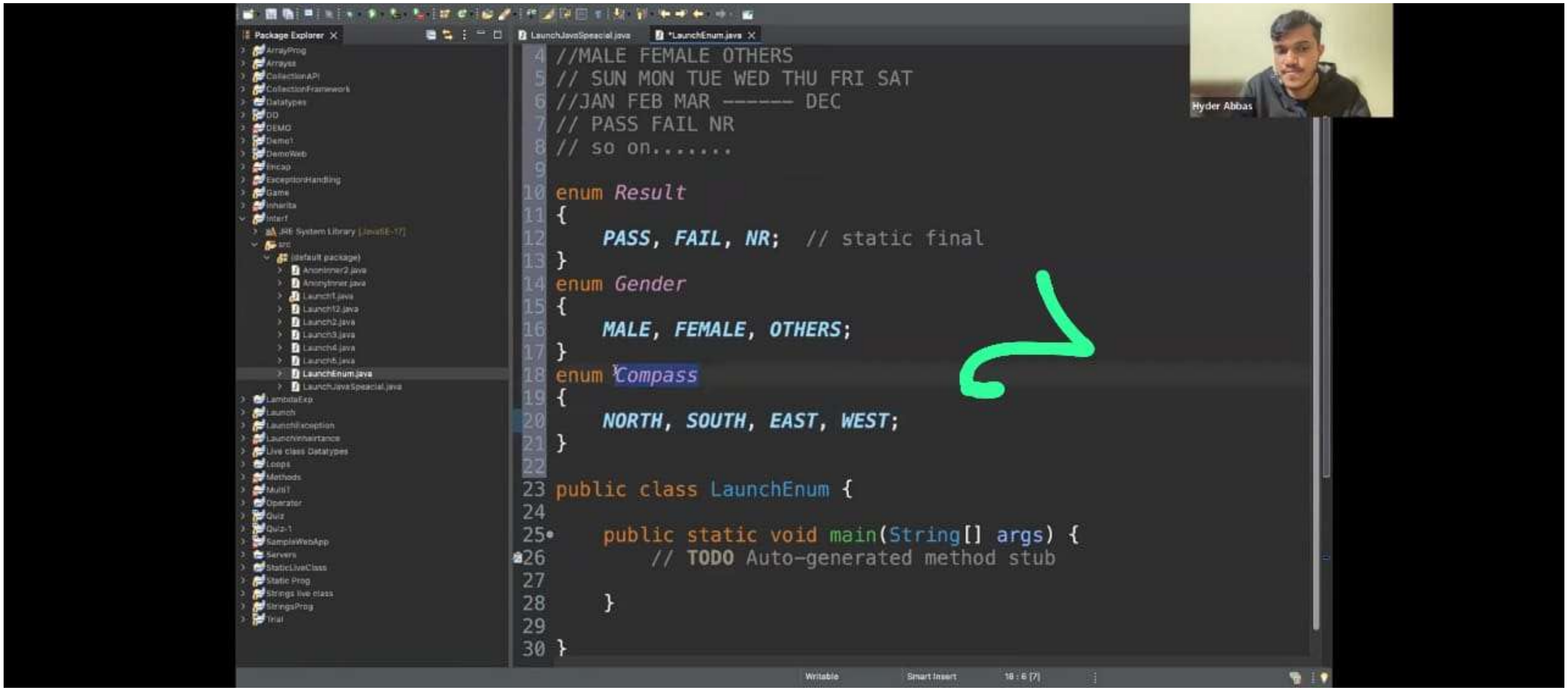
Java_Enum_And_Annotations

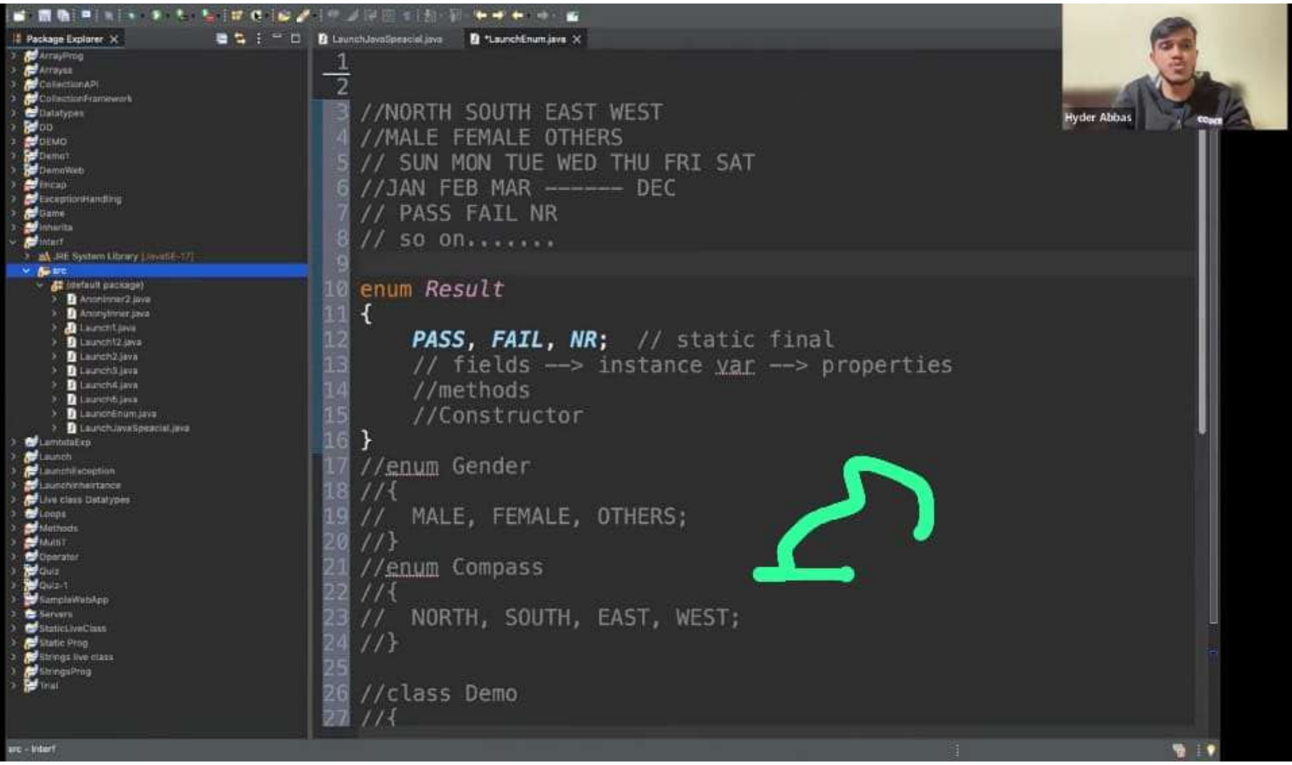


=> enum => group of predefined constant
-> enumerated data | our own Data type

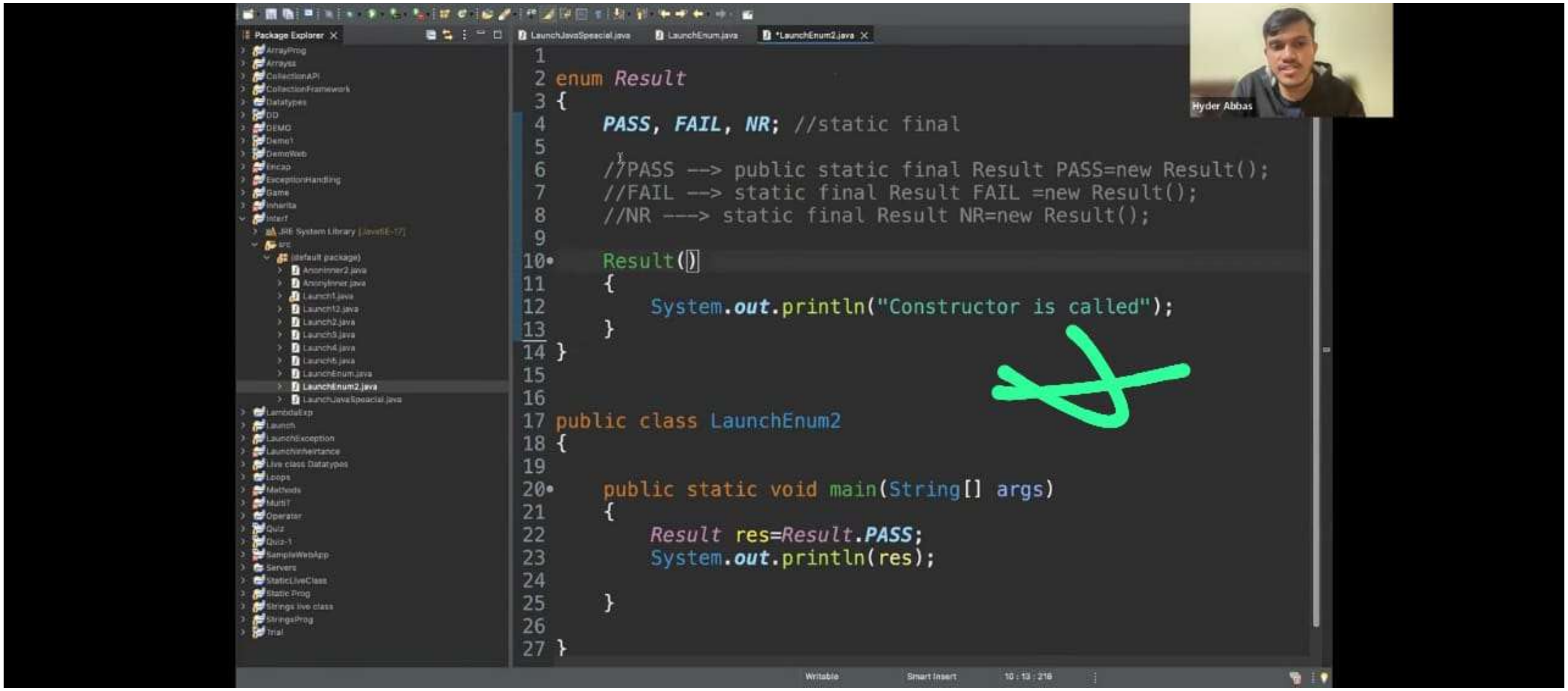
=> enum compass
{ NORTH SOUTH EAST WEST ;
}



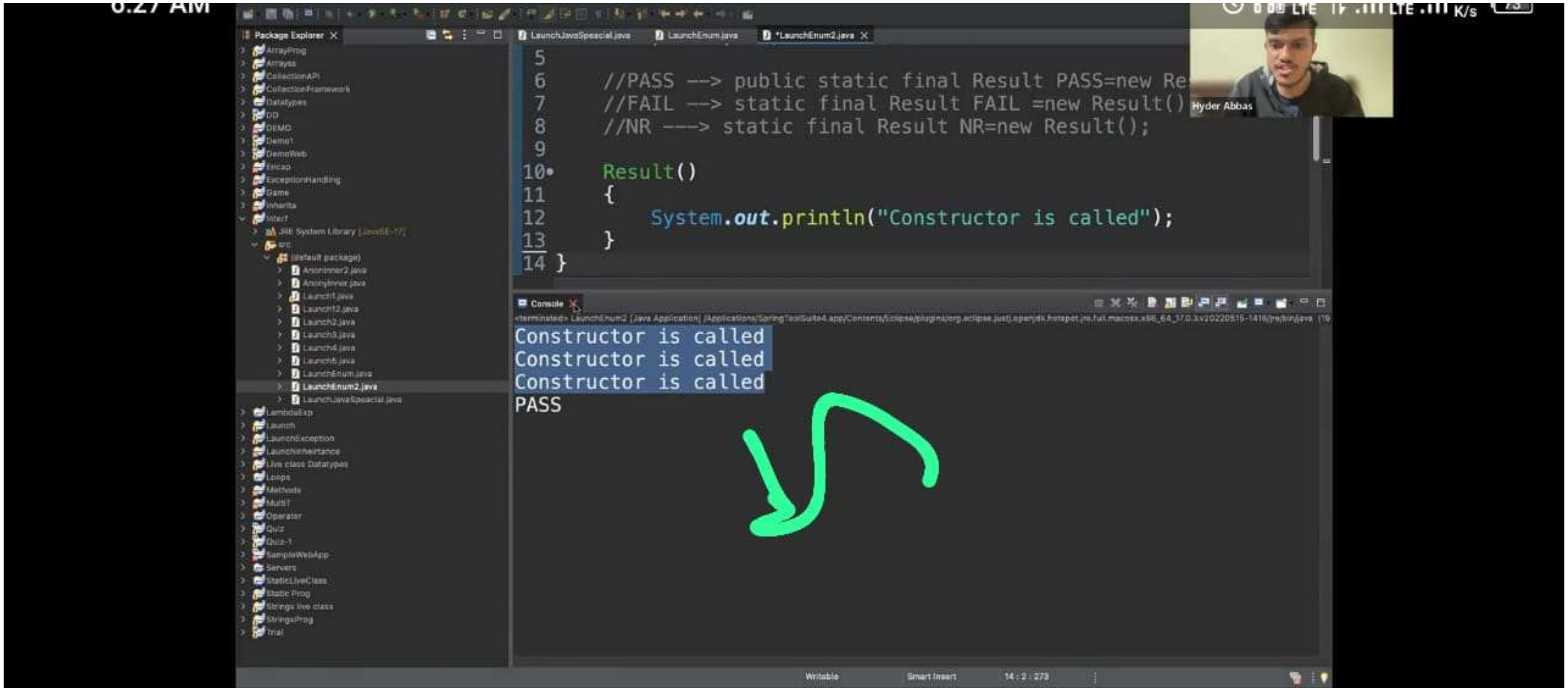
```java
//MALE FEMALE OTHERS
// SUN MON TUE WED THU FRI SAT
//JAN FEB MAR ------- DEC
// PASS FAIL NR
// so on.......

enum Result
{
    PASS, FAIL, NR;  // static final
}
enum Gender
{
    MALE, FEMALE, OTHERS;
}
enum Compass
{
    NORTH, SOUTH, EAST, WEST;
}

public class LaunchEnum {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}
```



```java


//NORTH SOUTH EAST WEST
//MALE FEMALE OTHERS
// SUN MON TUE WED THU FRI SAT
//JAN FEB MAR ------- DEC
// PASS FAIL NR
// so on.......

enum Result
{
    PASS, FAIL, NR;  // static final
    // fields --> instance var --> properties
    //methods
    //Constructor
}
//enum Gender
//{
//  MALE, FEMALE, OTHERS;
//}
//enum Compass
//{
//  NORTH, SOUTH, EAST, WEST;
//}

//class Demo
//{
```

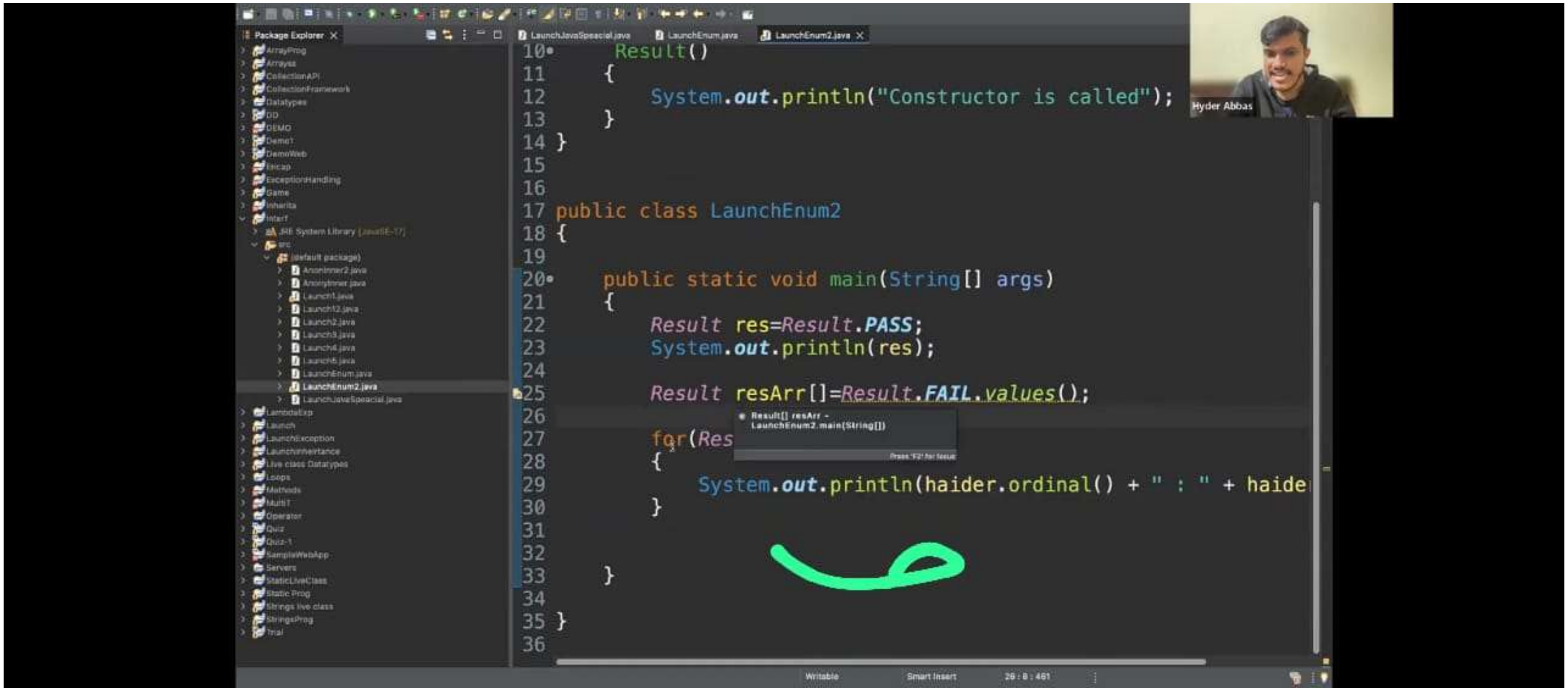


```java

enum Result
{
    PASS, FAIL, NR; //static final

    //PASS --> public static final Result PASS=new Result();
    //FAIL --> static final Result FAIL =new Result();
    //NR ---> static final Result NR=new Result();

    Result()
    {
        System.out.println("Constructor is called");
    }
}

public class LaunchEnum2
{

    public static void main(String[] args)
    {
        Result res=Result.PASS;
        System.out.println(res);

    }

}
```



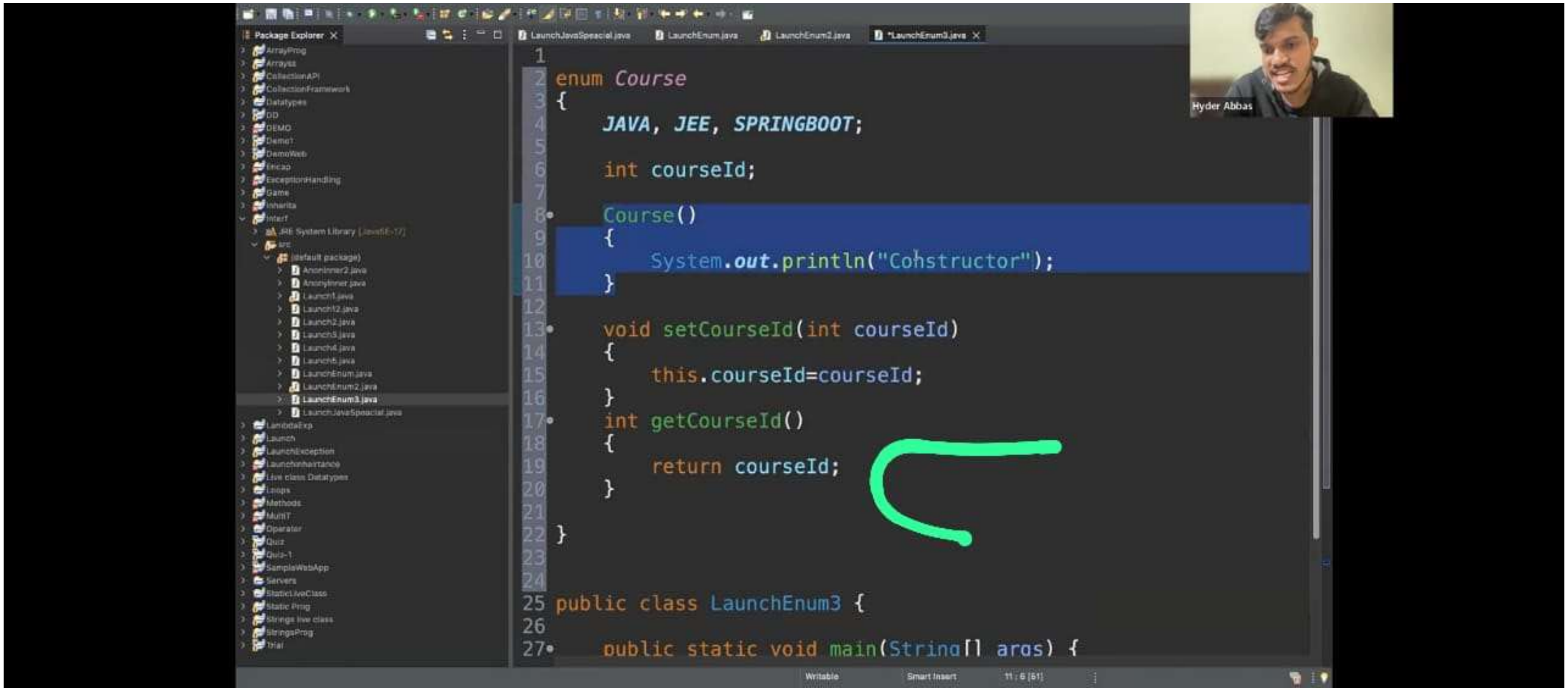
```

    //PASS --> public static final Result PASS=new Re
    //FAIL --> static final Result FAIL =new Result()
    //NR ---> static final Result NR=new Result();

    Result()
    {
        System.out.println("Constructor is called");
    }
}
```

```
<terminated> LaunchEnum2 [Java Application] /Applications/SpringToolSuite4.app/Contents/Eclipse/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_17.0.3.v20220515-1416/jre/bin/java (19
Constructor is called
Constructor is called
Constructor is called
PASS
```

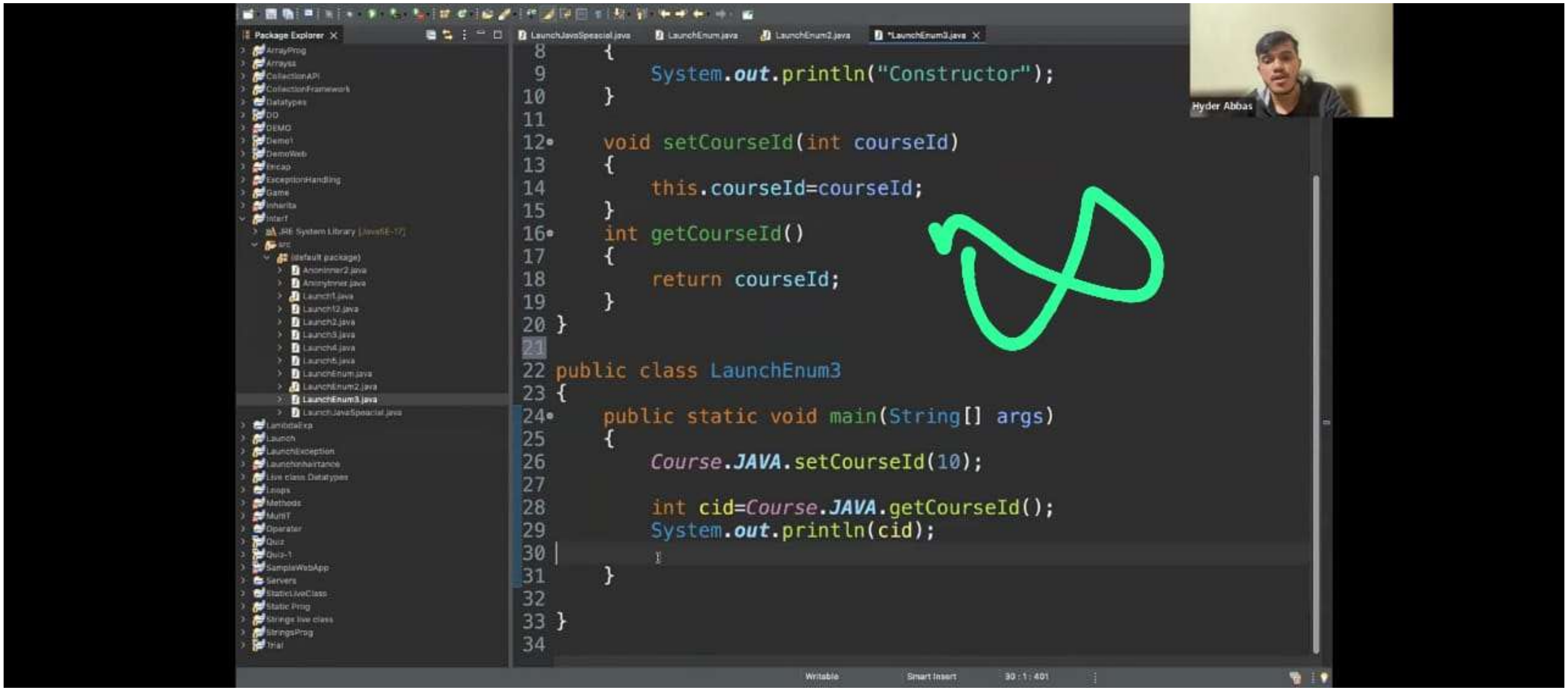


```java
     Result()
    {
        System.out.println("Constructor is called");
    }
}


public class LaunchEnum2
{

    public static void main(String[] args)
    {
        Result res=Result.PASS;
        System.out.println(res);

        Result resArr[]=Result.FAIL.values();

        for(Res
        {
            System.out.println(haider.ordinal() + " : " + haide
        }


    }

}
```

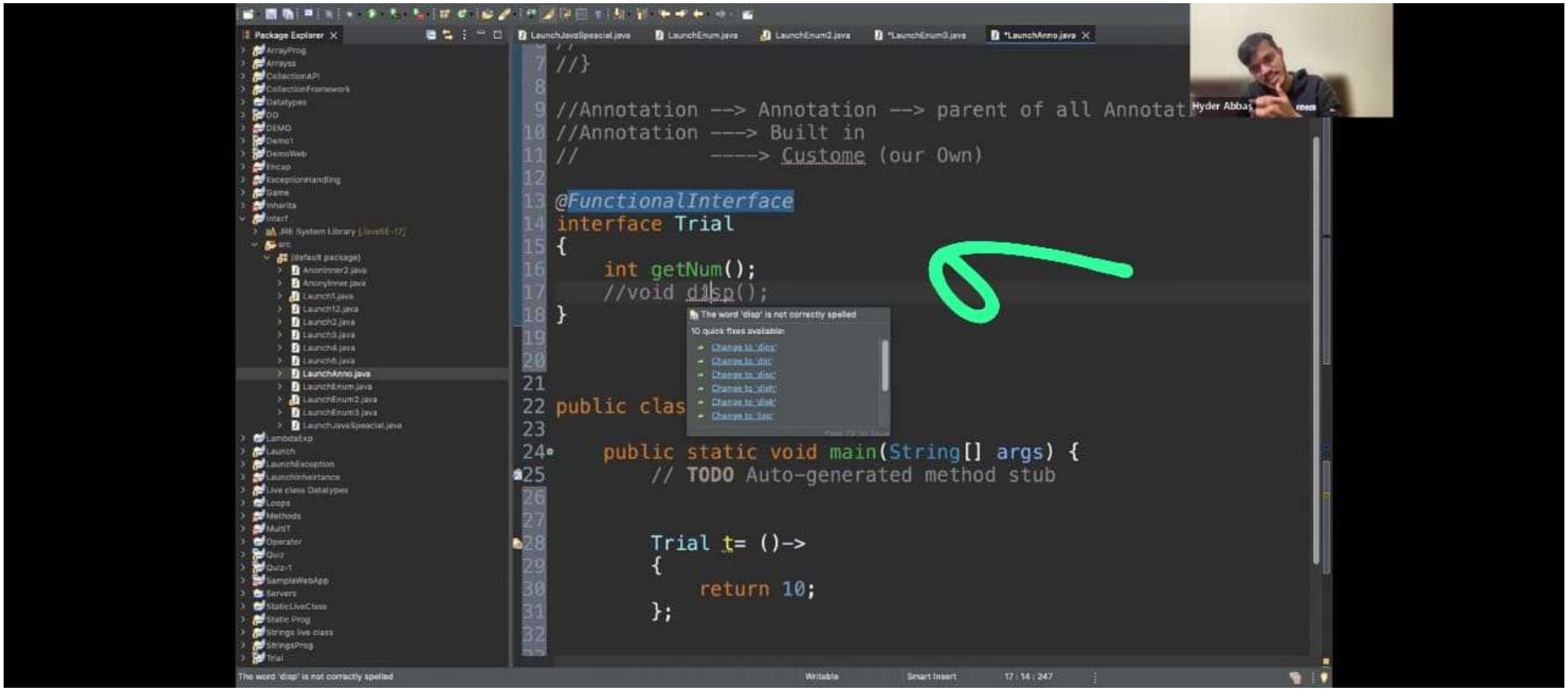
```
Result[] resArr - LaunchEnum2.main(String[])
Press 'F2' for focus
```



```java

enum Course
{
    JAVA, JEE, SPRINGBOOT;

    int courseId;

    Course()
    {
        System.out.println("Constructor");
    }

    void setCourseId(int courseId)
    {
        this.courseId=courseId;
    }
    int getCourseId()
    {
        return courseId;
    }

}


public class LaunchEnum3 {

    public static void main(String[] args) {
```



```java
    {
        System.out.println("Constructor");
    }

    void setCourseId(int courseId)
    {
        this.courseId=courseId;
    }
    int getCourseId()
    {
        return courseId;
    }
}

public class LaunchEnum3
{
    public static void main(String[] args)
    {
        Course.JAVA.setCourseId(10);

        int cid=Course.JAVA.getCourseId();
        System.out.println(cid);

    }

}
```

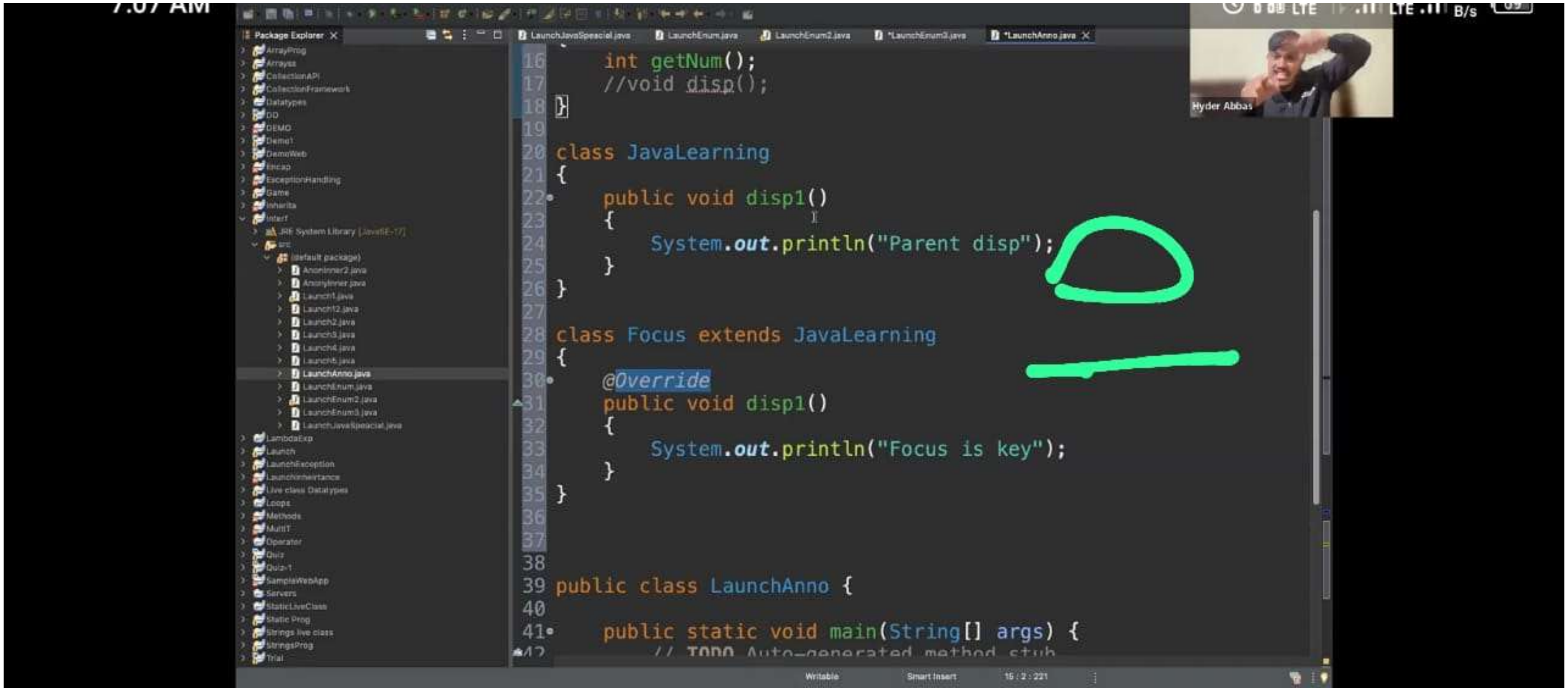


```
//}

//Annotation --> Annotation --> parent of all Annotat
//Annotation ---> Built in
//              ----> Custome (our Own)

@FunctionalInterface
interface Trial
{
    int getNum();
    //void disp();
}


public clas

    public static void main(String[] args) {
        // TODO Auto-generated method stub


        Trial t= ()->
        {
            return 10;
        };
```

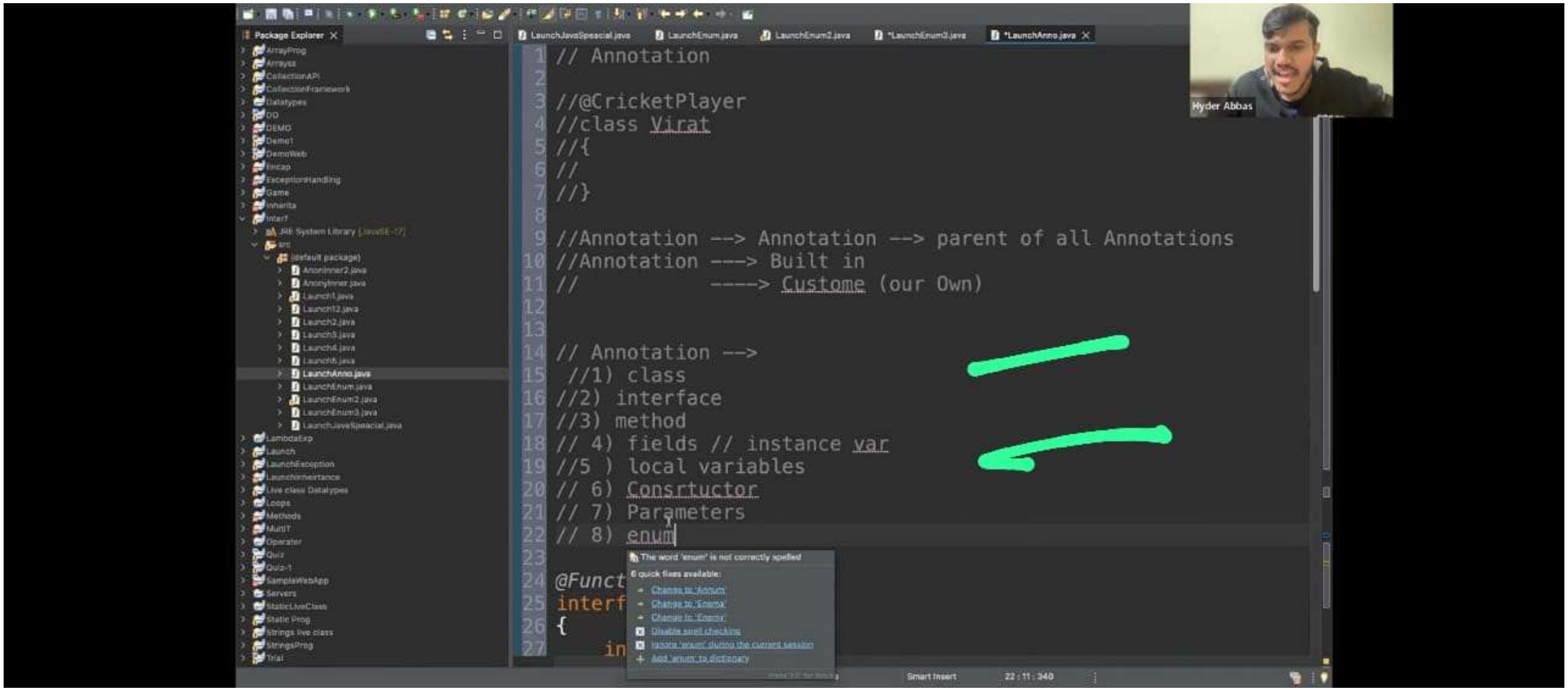
The word 'disp' is not correctly spelled

10 quick fixes available:

The word 'disp' is not correctly spelled



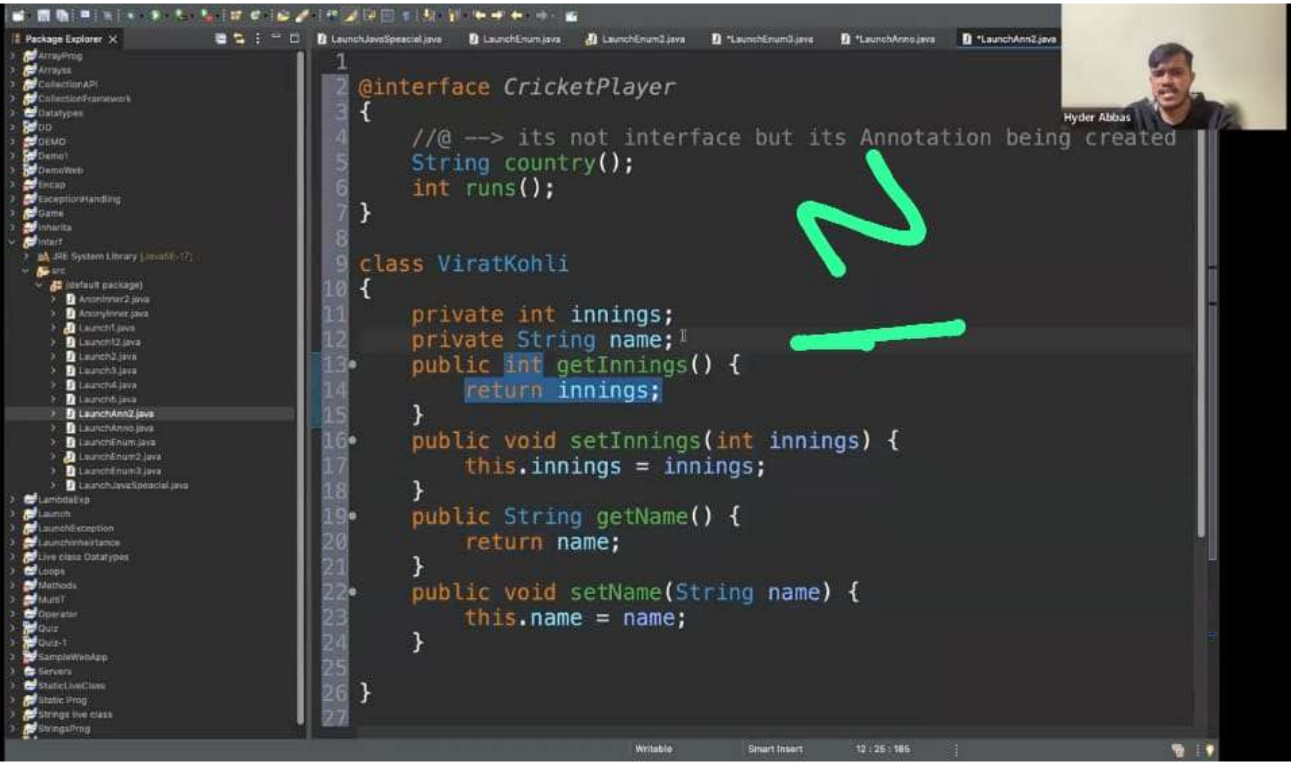
```
    int getNum();
    //void disp();
}

class JavaLearning
{
    public void disp1()
    {
        System.out.println("Parent disp");
    }
}

class Focus extends JavaLearning
{
    @Override
    public void disp1()
    {
        System.out.println("Focus is key");
    }
}


public class LaunchAnno {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
```



```
// Annotation

//@CricketPlayer
//class Virat
//{
//
//}

//Annotation --> Annotation --> parent of all Annotations
//Annotation ---> Built in
//              ----> Custome (our Own)


// Annotation -->
 //1) class
//2) interface
//3) method
// 4) fields // instance var
//5 ) local variables
// 6) Consrtuctor
// 7) Parameters
// 8) enum

@Funct
interf
{
    in
```

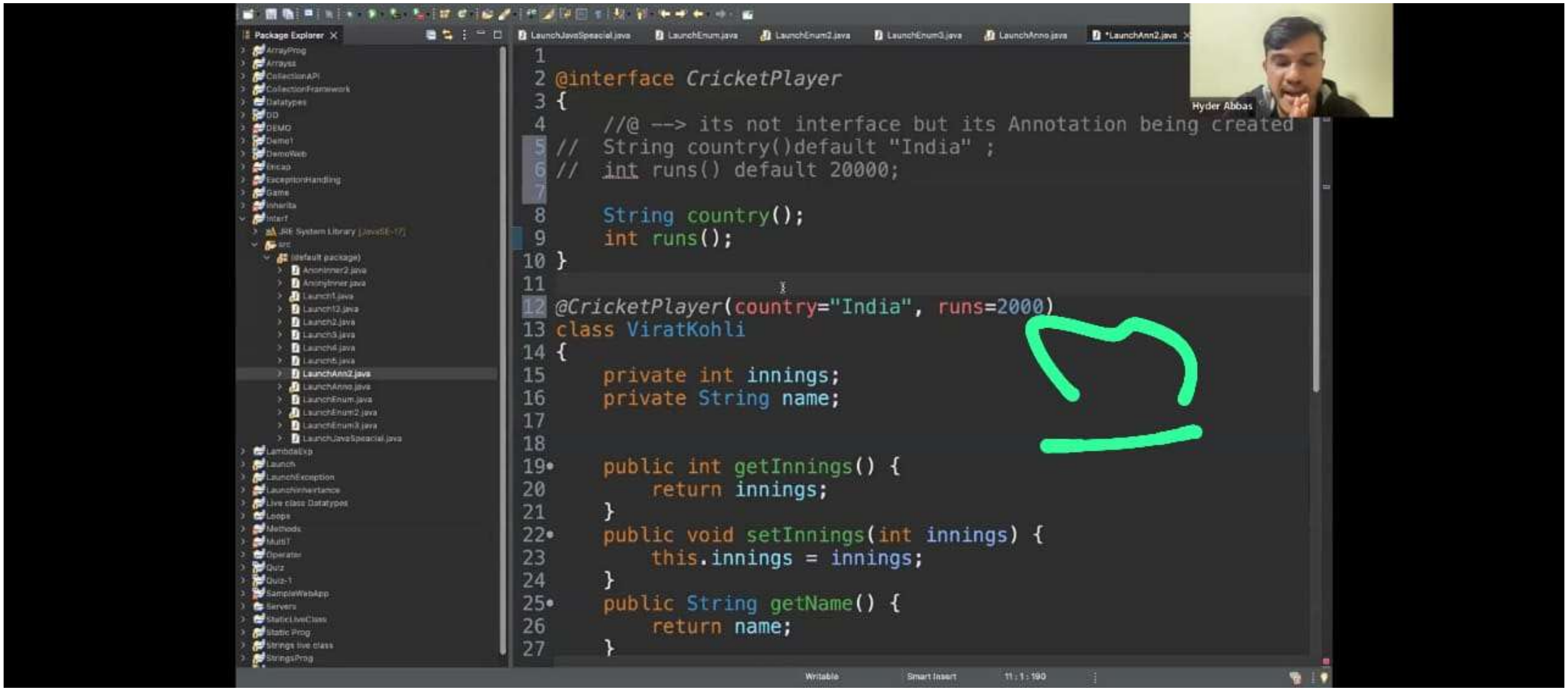
The word 'enum' is not correctly spelled

6 quick fixes available:

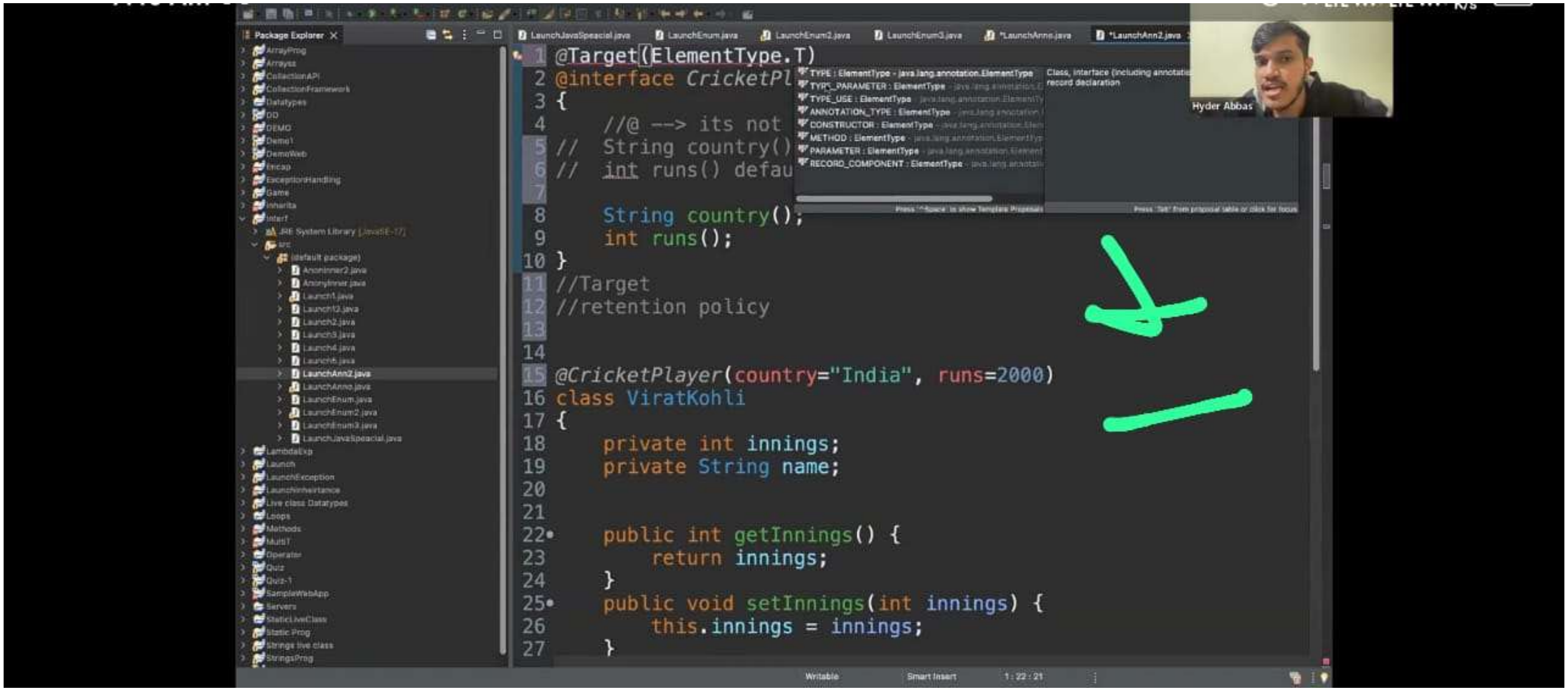


```java

@interface CricketPlayer
{
    //@ --> its not interface but its Annotation being created
    String country();
    int runs();
}

class ViratKohli
{
    private int innings;
    private String name;
    public int getInnings() {
        return innings;
    }
    public void setInnings(int innings) {
        this.innings = innings;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

}
```

12



```java

@interface CricketPlayer
{
    //@ --> its not interface but its Annotation being created
//  String country()default "India" ;
//  int runs() default 20000;

    String country();
    int runs();
}

@CricketPlayer(country="India", runs=2000)
class ViratKohli
{
    private int innings;
    private String name;


    public int getInnings() {
        return innings;
    }
    public void setInnings(int innings) {
        this.innings = innings;
    }
    public String getName() {
        return name;
    }
```



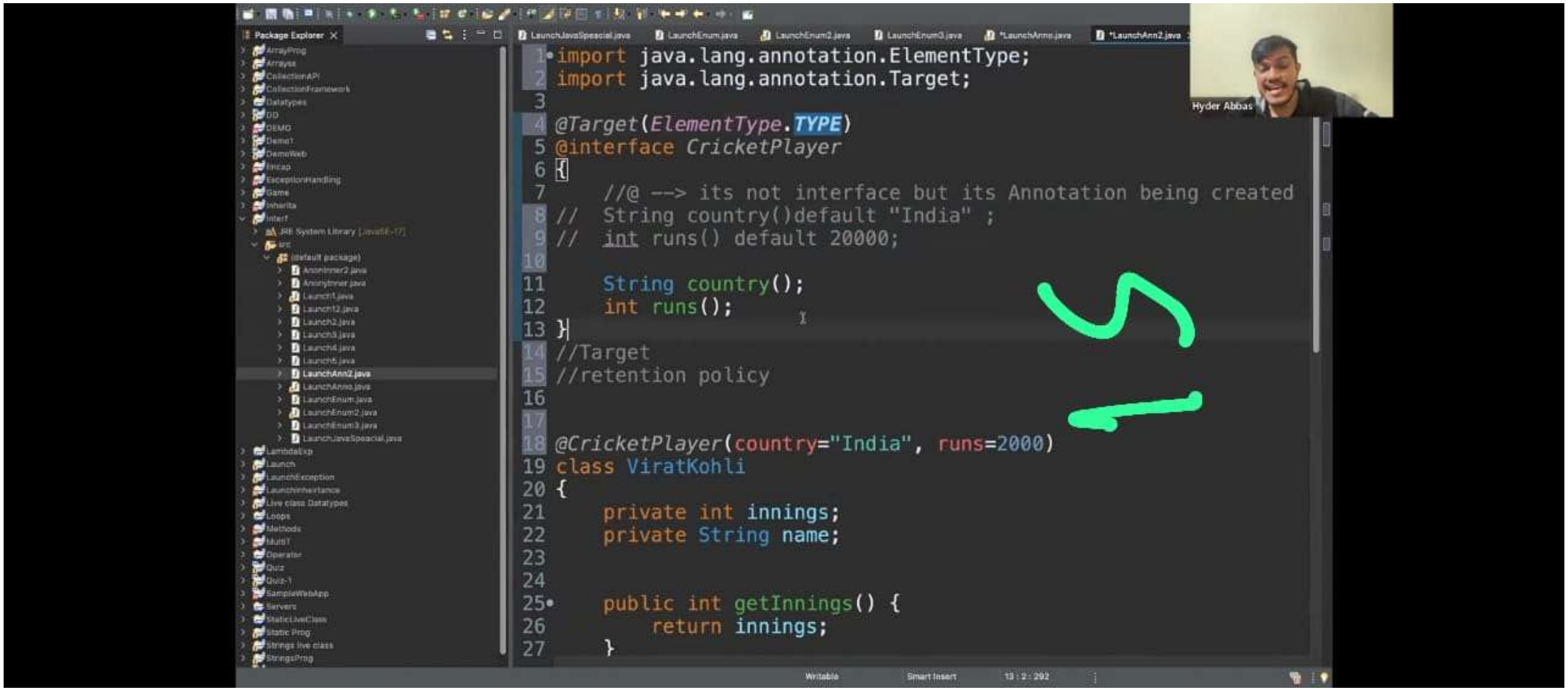
```java
@Target(ElementType.T)
@interface CricketPl
{
    //@ --> its not
//  String country()
//  int runs() defau

    String country();
    int runs();
}
//Target
//retention policy


@CricketPlayer(country="India", runs=2000)
class ViratKohli
{
    private int innings;
    private String name;


    public int getInnings() {
        return innings;
    }
    public void setInnings(int innings) {
        this.innings = innings;
    }
```

```
TYPE : ElementType - java.lang.annotation.ElementType
TYPE_PARAMETER : ElementType - java.lang.annotation.E
TYPE_USE : ElementType - java.lang.annotation.ElementTy
ANNOTATION_TYPE : ElementType - java.lang.annotation.
CONSTRUCTOR : ElementType - java.lang.annotation.Elem
METHOD : ElementType - java.lang.annotation.ElementTyp
PARAMETER : ElementType - java.lang.annotation.Element
RECORD_COMPONENT : ElementType - java.lang.annotati
```

Class, interface (including annotati
record declaration

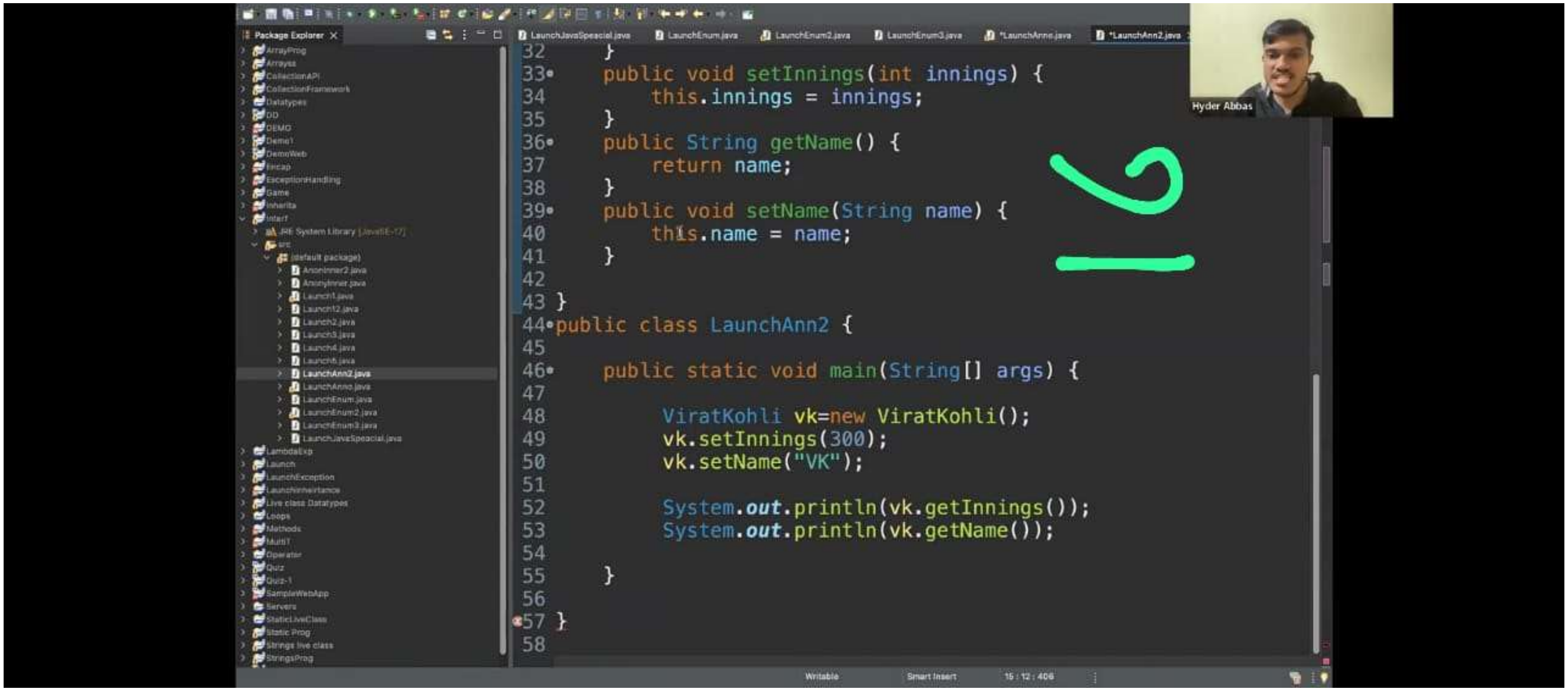
Press '^Space' to show Template Proposals

Press 'Tab' from proposal table or click for focus

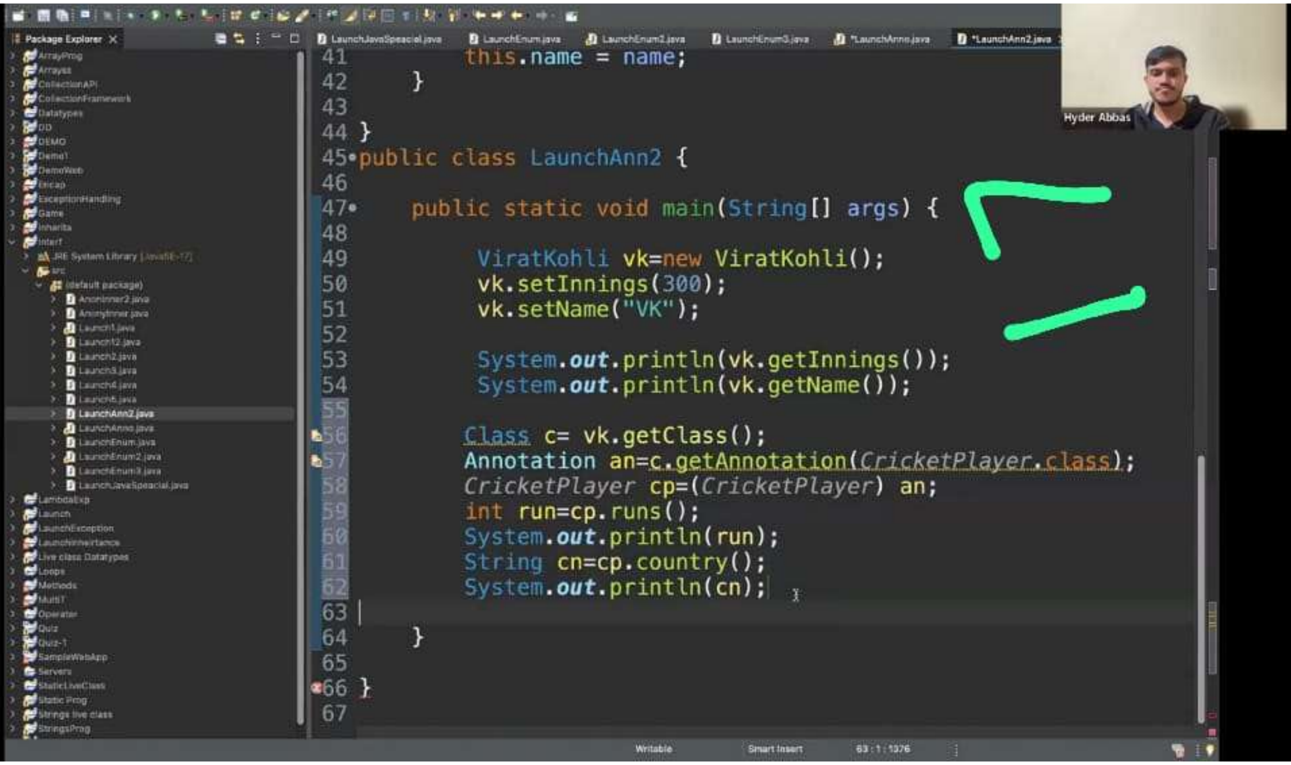


```java
import java.lang.annotation.ElementType;
import java.lang.annotation.Target;

@Target(ElementType.TYPE)
@interface CricketPlayer
{
    //@ --> its not interface but its Annotation being created
//  String country()default "India" ;
//  int runs() default 20000;

    String country();
    int runs();
}
//Target
//retention policy


@CricketPlayer(country="India", runs=2000)
class ViratKohli
{
    private int innings;
    private String name;


    public int getInnings() {
        return innings;
    }
```

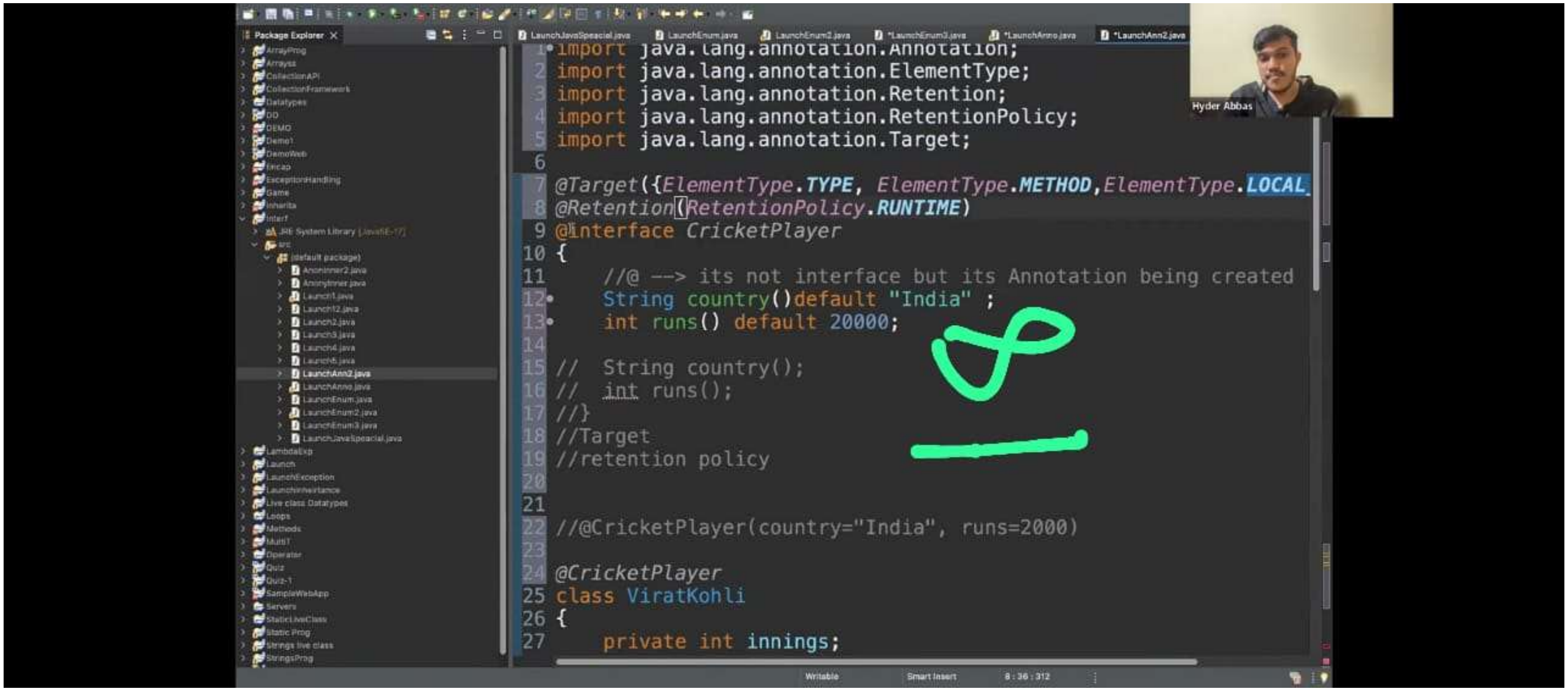
15



```java
    }
    public void setInnings(int innings) {
        this.innings = innings;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

}
public class LaunchAnn2 {

    public static void main(String[] args) {

         ViratKohli vk=new ViratKohli();
         vk.setInnings(300);
         vk.setName("VK");

         System.out.println(vk.getInnings());
         System.out.println(vk.getName());

    }

}

```

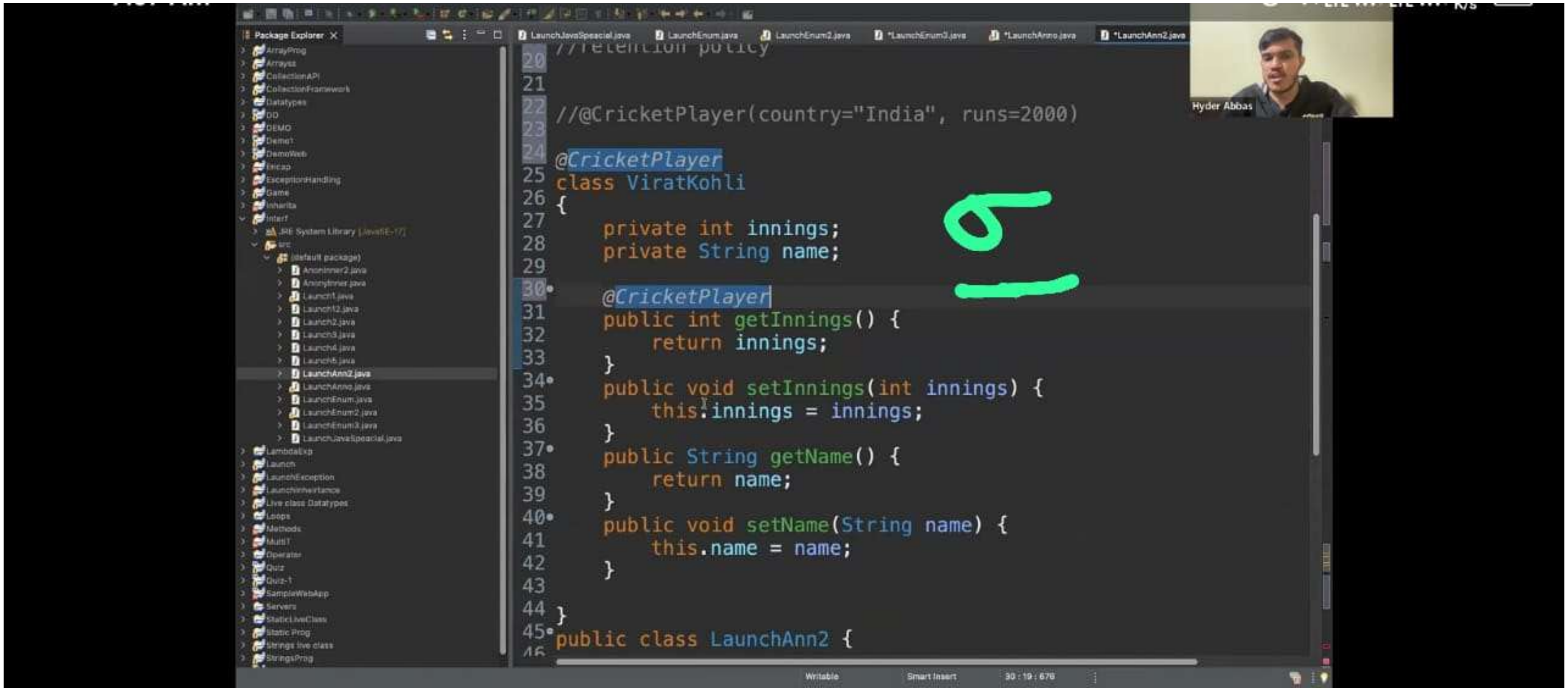


```java
        this.name = name;
    }

}
public class LaunchAnn2 {

    public static void main(String[] args) {

        ViratKohli vk=new ViratKohli();
        vk.setInnings(300);
        vk.setName("VK");

        System.out.println(vk.getInnings());
        System.out.println(vk.getName());

        Class c= vk.getClass();
        Annotation an=c.getAnnotation(CricketPlayer.class);
        CricketPlayer cp=(CricketPlayer) an;
        int run=cp.runs();
        System.out.println(run);
        String cn=cp.country();
        System.out.println(cn);

    }

}
```



```java
import java.lang.annotation.Annotation;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Target({ElementType.TYPE, ElementType.METHOD,ElementType.LOCAL_
@Retention(RetentionPolicy.RUNTIME)
@interface CricketPlayer
{
    //@ --> its not interface but its Annotation being created
    String country()default "India" ;
    int runs() default 20000;

//  String country();
//  int runs();
//}
//Target
//retention policy


//@CricketPlayer(country="India", runs=2000)

@CricketPlayer
class ViratKohli
{
    private int innings;
```

18



```
//retention policy

//@CricketPlayer(country="India", runs=2000)

@CricketPlayer
class ViratKohli
{
    private int innings;
    private String name;

    @CricketPlayer
    public int getInnings() {
        return innings;
    }
    public void setInnings(int innings) {
        this.innings = innings;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

}
public class LaunchAnn2 {
```