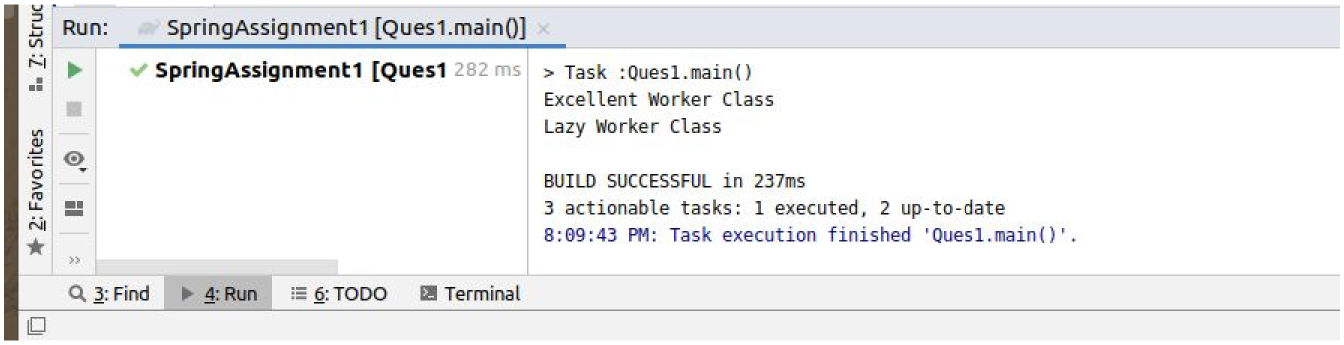
Session : Spring Framework

(1) Write a program to demonstrate Tightly Coupled code.

CODE

```
public class Ques1 {
  public static void main(String[] args) {
     ExcellentWorker excellent = new ExcellentWorker();
     LazyWorker lazy = new LazyWorker();
     excellent.dowork();
     lazy.dowork();
  }
}


public interface Worker {
  void dowork();
}


public class ExcellentWorker implements Worker{
  @Override
  public void dowork() {
        System.out.println("Excellent Worker Class");
  }
}

public class LazyWorker implements Worker{
  @Override
  public void dowork() {
        System.out.println("Lazy Worker Class");
  }
}
```

OUTPUT



Run:
SpringAssignment1 [Ques1.main()]
SpringAssignment1 [Ques1 282 ms
3: Find
4: Run
6: TODO
Terminal
7: Struc
2: Favorites

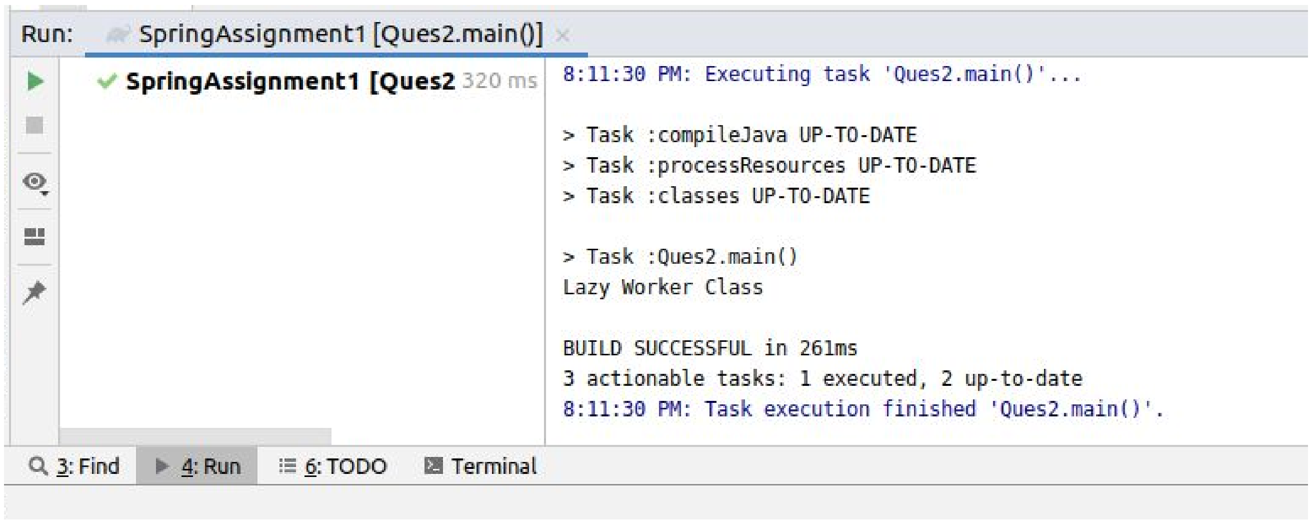
```
> Task :Ques1.main()
Excellent Worker Class
Lazy Worker Class

BUILD SUCCESSFUL in 237ms
3 actionable tasks: 1 executed, 2 up-to-date
8:09:43 PM: Task execution finished 'Ques1.main()'.
```

(2) Write a program to demonstrate Loosely Coupled code.

CODE

```
public class Ques2 {
  public static void main(String[] args)
  {
     new Manager2(new LazyWorker2()).work();
  }
}

public class Manager2 {
  Worker2 worker2;
  Manager2(Worker2 worker2)
  {
     this.worker2=worker2;
  }


  void work(){
     worker2.dowork();
  }
}


public interface Worker2 {
  void dowork();
}


public class ExcellentWorker2 implements Worker2{
  @Override
  public void dowork() {
        System.out.println("Excellent Worker Class");
    }
}


public class LazyWorker2 implements Worker2{
    @Override
    public void dowork() {
        System.out.println("Lazy Worker Class");
    }
}
```

OUTPUT


Run:
SpringAssignment1 [Ques2.main()]
SpringAssignment1 [Ques2 320 ms
3: Find
4: Run
6: TODO
Terminal

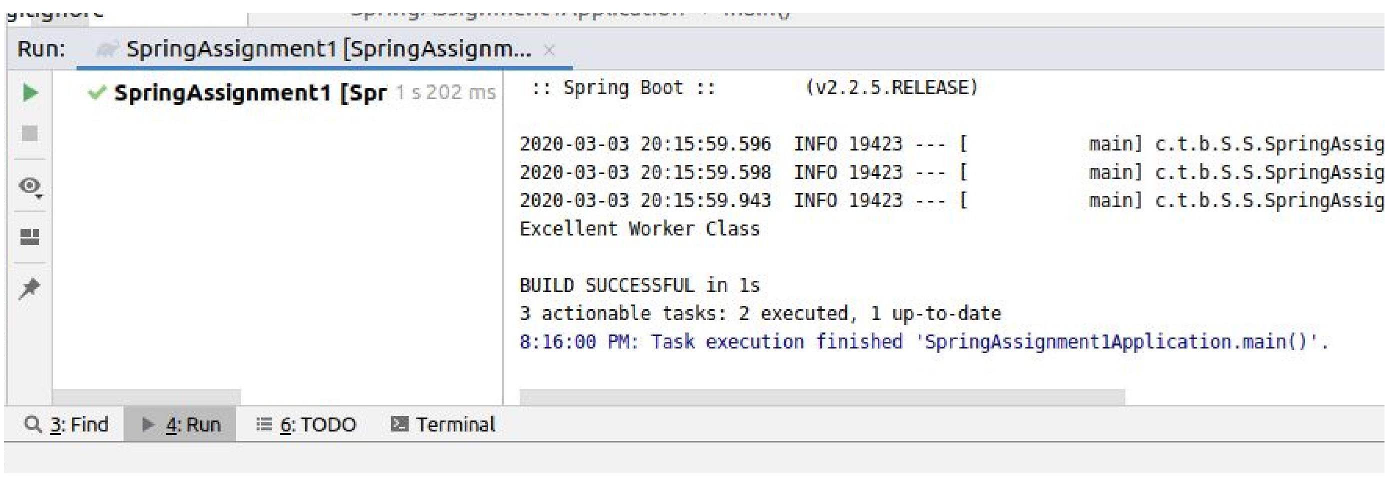
```
8:11:30 PM: Executing task 'Ques2.main()'...

> Task :compileJava UP-TO-DATE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE

> Task :Ques2.main()
Lazy Worker Class

BUILD SUCCESSFUL in 261ms
3 actionable tasks: 1 executed, 2 up-to-date
8:11:30 PM: Task execution finished 'Ques2.main()'.
```

(3) Use @Compenent and @Autowired annotations to in Loosely Coupled code for dependency management.

CODE

```
@SpringBootApplication
public class SpringAssignment1Application {

  public static void main(String[] args) {
     ApplicationContext applicationContext3 =
SpringApplication.run(SpringAssignment1Application.class, args);

//    Ques3
     Manager3 manager3 = applicationContext3.getBean(Manager3.class);
     manager3.managework();
            }
}


@Component
public class Manager3 {
  @Autowired
  public Worker3 worker3;

  Manager3(Worker3 worker3)
  {
     this.worker3 =worker3;
  }

  public void managework(){
     worker3.dowork();
  }
}


public interface Worker3 {
  void dowork();
}


@Component
public class ExcellentWorker3 implements Worker3{
  @Override
  public void dowork() {
     System.out.println("Excellent Worker Class");
  }
}


public class LazyWorker3 implements Worker3{
  @Override
  public void dowork() {
     System.out.println("Lazy Worker Class");
  }
}
```

OUTPUT


Run:
SpringAssignment1 [SpringAssignm...
SpringAssignment1 [Spr 1 s 202 ms
3: Find
4: Run
6: TODO
Terminal

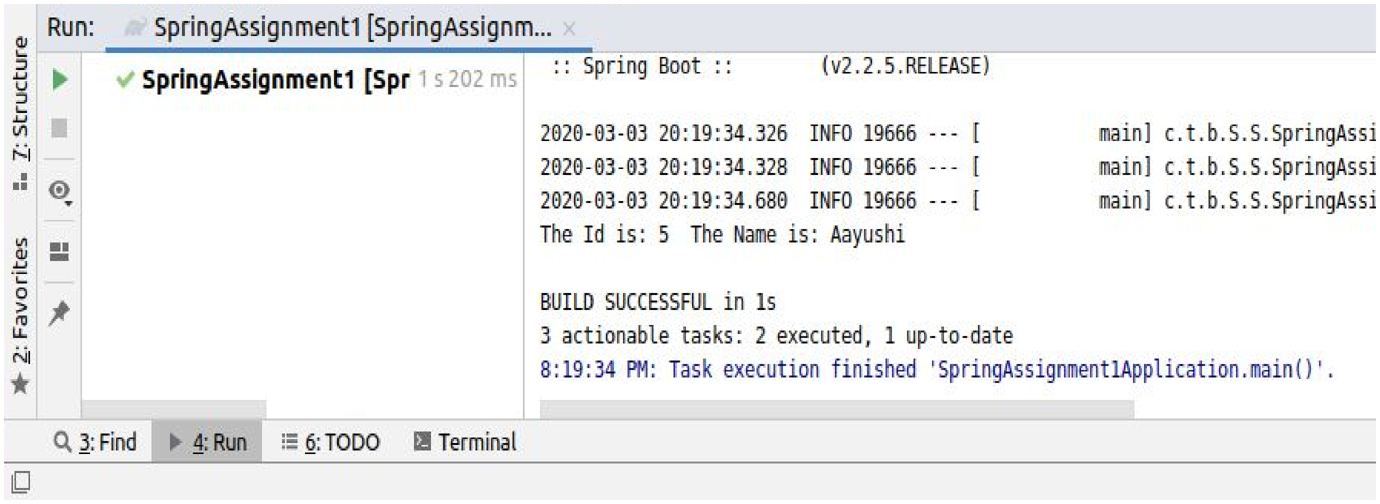
```
 :: Spring Boot ::        (v2.2.5.RELEASE)

2020-03-03 20:15:59.596  INFO 19423 --- [           main] c.t.b.S.S.SpringAssig
2020-03-03 20:15:59.598  INFO 19423 --- [           main] c.t.b.S.S.SpringAssig
2020-03-03 20:15:59.943  INFO 19423 --- [           main] c.t.b.S.S.SpringAssig
Excellent Worker Class

BUILD SUCCESSFUL in 1s
3 actionable tasks: 2 executed, 1 up-to-date
8:16:00 PM: Task execution finished 'SpringAssignment1Application.main()'.
```

(4) Get a Spring Bean from application context and display its properties.

CODE

```
@SpringBootApplication
public class SpringAssignment1Application {

  public static void main(String[] args) {
     ApplicationContext applicationContext3 =
SpringApplication.run(SpringAssignment1Application.class, args);
     //Ques4
     Manager4 manager4 = applicationContext3.getBean(Manager4.class);
     manager4.managework();
}
}

@Component
public class Manager4 {

   @Autowired
   public Worker4 worker4;

   Manager4(Worker4 worker4)
   {
       this.worker4 =worker4;
   }
  public void managework(){
     worker4.show();
  }
}


public interface Worker4 {
  void dowork();

  void show();
}

@Component
public class ExcellentWorker4 implements Worker4{
  private int id = 5;
  private String name = "Aayushi";

  @Override
  public void dowork() {
     System.out.println("Excellent Worker Class");
  }

  public void show(){
System.out.println("The Id is: "+id+ "The Name is: "+name);
  }
}

public class LazyWorker4 implements Worker4{
  @Override
  public void dowork() {
     System.out.println("Lazy Worker Class");
  }

  @Override
  public void show() {

  }
}
```

OUTPUT


Run:
SpringAssignment1 [SpringAssignm...
SpringAssignment1 [Spr 1 s 202 ms
7: Structure
2: Favorites
3: Find
4: Run
6: TODO
Terminal

```
 :: Spring Boot ::        (v2.2.5.RELEASE)

2020-03-03 20:19:34.326  INFO 19666 --- [           main] c.t.b.S.S.SpringAssi
2020-03-03 20:19:34.328  INFO 19666 --- [           main] c.t.b.S.S.SpringAssi
2020-03-03 20:19:34.680  INFO 19666 --- [           main] c.t.b.S.S.SpringAssi
The Id is: 5  The Name is: Aayushi

BUILD SUCCESSFUL in 1s
3 actionable tasks: 2 executed, 1 up-to-date
8:19:34 PM: Task execution finished 'SpringAssignment1Application.main()'.
```

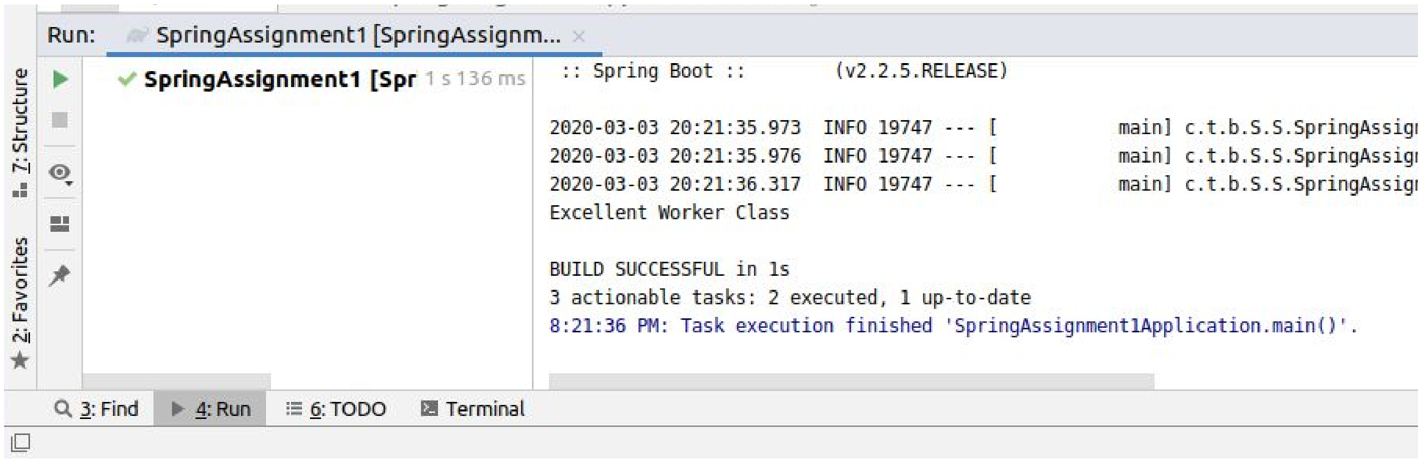
(5) Demonstrate how you will resolve ambiguity while autowiring bean (Hint : @Primary)

CODE

```
@SpringBootApplication
public class SpringAssignment1Application {

  public static void main(String[] args) {
     ApplicationContext applicationContext3 =
SpringApplication.run(SpringAssignment1Application.class, args);
     //Ques5
     Manager5 manager5 = applicationContext3.getBean(Manager5.class);
     manager5.managework();
}
}
@Component
public class Manager5 {
   @Autowired
   public Worker5 worker5;

   Manager5(Worker5 worker5)
   {
       this.worker5 =worker5;
   }

   public void managework(){
       worker5.dowork();
   }
}
```

```
public interface Worker5 {
  void dowork();
}

@Component
@Primary
public class ExcellentWorker5 implements Worker5{
  @Override
  public void dowork() {

     System.out.println("Excellent Worker Class");
  }
}

@Component
public class LazyWorker5 implements Worker5{
  @Override
  public void dowork() {
     System.out.println("Lazy Worker Class");
  }
}
```

OUTPUT


Run:
SpringAssignment1 [SpringAssignm...
SpringAssignment1 [Spr 1 s 136 ms
7: Structure
2: Favorites
3: Find
4: Run
6: TODO
Terminal

```
 :: Spring Boot ::        (v2.2.5.RELEASE)

2020-03-03 20:21:35.973  INFO 19747 --- [           main] c.t.b.S.S.SpringAssig
2020-03-03 20:21:35.976  INFO 19747 --- [           main] c.t.b.S.S.SpringAssig
2020-03-03 20:21:36.317  INFO 19747 --- [           main] c.t.b.S.S.SpringAssig
Excellent Worker Class

BUILD SUCCESSFUL in 1s
3 actionable tasks: 2 executed, 1 up-to-date
8:21:36 PM: Task execution finished 'SpringAssignment1Application.main()'.
```

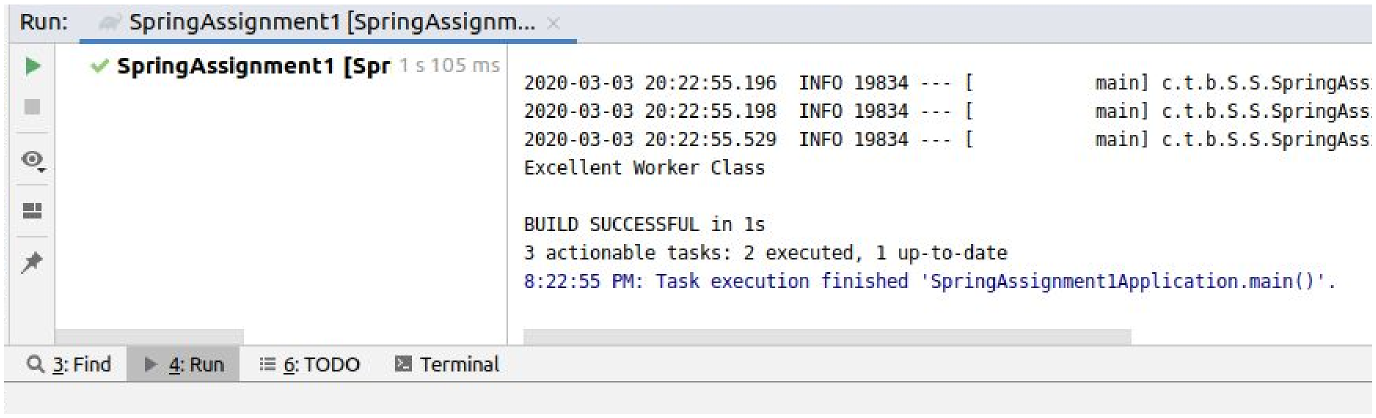
(6) Perform Constructor Injection in a Spring Bean.

CODE

```
@SpringBootApplication
```

```
public class SpringAssignment1Application {

  public static void main(String[] args) {
     ApplicationContext applicationContext3 =
SpringApplication.run(SpringAssignment1Application.class, args);

     //Ques6
     Manager6 manager6 = applicationContext3.getBean(Manager6.class);
     manager6.managework();
  }
}

@Component
public class Manager6 {
   public Worker6 worker6;

   @Autowired
   Manager6(Worker6 worker6)
   {
       this.worker6 =worker6;
   }

   public void managework(){
       worker6.dowork();
   }
}
ublic interface Worker6 {
   void dowork();
}


@Component
@Primary
public class ExcellentWorker6 implements Worker6 {
   @Override
   public void dowork() {
       System.out.println("Excellent Worker Class");
   }
}


@Component
public class LazyWorker6 implements Worker6{
   @Override
   public void dowork() {
       System.out.println("Lazy Worker Class");
   }
}
```

OUTPUT



Run:
SpringAssignment1 [SpringAssignm...
SpringAssignment1 [Spr 1 s 105 ms
3: Find
4: Run
6: TODO
Terminal

```
2020-03-03 20:22:55.196  INFO 19834 --- [           main] c.t.b.S.S.SpringAss
2020-03-03 20:22:55.198  INFO 19834 --- [           main] c.t.b.S.S.SpringAss
2020-03-03 20:22:55.529  INFO 19834 --- [           main] c.t.b.S.S.SpringAss
Excellent Worker Class

BUILD SUCCESSFUL in 1s
3 actionable tasks: 2 executed, 1 up-to-date
8:22:55 PM: Task execution finished 'SpringAssignment1Application.main()'.
```