Simple.java

```
import java.lang.annotation.*;

@Retention(RetentionPolicy.RUNTIME)
@interface Author{
   String name();
   String date();
}

@Retention(RetentionPolicy.RUNTIME)
@interface Descriptor{
   String str();
}

@Author(name="John Doe", date="2024-06-01")
class Example {
   @Descriptor(str="This is a simple class with custom annotations.")
   public void getData(){
      System.out.println("HI");
   }
}

public class Simple {
  public static void main(String[] args) {
      Class<Example> obj = Example.class;

      Author authorAnnotation = obj.getAnnotation(Author.class);
```

```
        System.out.println("Author: " + authorAnnotation.name() + ", Date: " +
authorAnnotation.date());


        System.out.println("All annotrations:");
        for (Annotation annotation : obj.getAnnotations()) {
            System.out.println(annotation);
        }
    }
}
```

Method Annotation

```
public class AnnotMethod {
   @Author(name="John Doe", date="2024-06-01")
   private void exMethod1() {
      // Method implementation
   }

   @Info(str="This method has a custom annotation.")
   public void exMethod2() {
      // Method implementation
   }

   @Info(str="This method has another custom annotation.")
   @Descriptor(str="This method is also annotated with a descriptor.")
   private void exMethod3(String param,int number) {
      System.out.println("Invoked private "+param+" with number "+number);
   }

   public static void main(String[] args) throws IllegalAccessException,
InvocationTargetException {
      System.out.println(AnnotMethod.class.getDeclaredAnnotation(null));
      Class<AnnotMethod> obj = AnnotMethod.class;
      try {
         System.out.println("Annotations for exMethod1:");
         Method exMethod1 = obj.getDeclaredMethod("exMethod1");
         exMethod1.setAccessible(true);
         exMethod1.invoke(new AnnotMethod());
         for(Annotation annotation : exMethod1.getAnnotations()) {
```

```
            System.out.println(annotation);
        }

        System.out.println("\nAnnotations for exMethod3:");
        Method exMethod3 = obj.getDeclaredMethod("exMethod3", String.class,
int.class);
        for(Annotation annotation : exMethod3.getAnnotations()) {
            System.out.println(annotation);
        }

        exMethod3.setAccessible(true);
        exMethod3.invoke(new AnnotMethod(), "Hello", 42);

      } catch (Exception e) {
        e.printStackTrace();
      }
   }
}
```

getAnnotationByType()

```
@interface Single {
   String val();
}

@Inherited
@Retention(RetentionPolicy.RUNTIME)
@interface A{
   String name();
   String date();
}

@Inherited
@Retention(RetentionPolicy.RUNTIME)
@interface Desc{
   String str();
}

@Retention(RetentionPolicy.RUNTIME)
@interface Inf{
   String str();
}

@A(name="John Doe", date="2024-06-01")
@Desc(str="This is a parent class with custom annotations.")
@Inf(str="This annotation is not inherited.")
class Base{
   // Parent class implementation
```

```
}

@A(name="Jane Smith", date="2024-06-02")
@Desc(str="This is a child class with custom annotations.")
@Inf(str="This annotation is not inherited.")
class Sub extends Base{
   // Child class implementation
}

public class AnnotType {
   @SuppressWarnings("all")
   public static void main(String[] args) {
      for(Annotation annotation : Sub.class.getAnnotationsByType(Inf.class)) {
         System.out.println(annotation);
      }


      Integer i = new Integer(10);
      System.out.println(i);
   }
}
```

getDeclaredAnnotation

```
@Inherited
@Retention(RetentionPolicy.RUNTIME)
@interface Autho{
   String name();
   String date();
}

@Inherited
@Retention(RetentionPolicy.RUNTIME)
@interface Descripto{
   String str();
}

@Retention(RetentionPolicy.RUNTIME)
@interface Info{
   String str();
}

@Autho(name="John Doe", date="2024-06-01")
@Descripto(str="This is a parent class with custom annotations.")
@Info(str="This annotation is not inherited.")
class Parent{
   // Parent class implementation
}

@Autho(name="Jane Smith", date="2024-06-02")
@Info(str="This annotation is not inherited.")
```

```
class Child extends Parent{
   // Child class implementation
}

public class Inherit {
   public static void main(String[] args) {
      Class<Child> obj = Child.class;
      for (Annotation annotation : obj.getDeclaredAnnotations()) {
         System.out.println(annotation);
      }
   }
}
```