9. ABSTRACT CLASS

1.

```
abstract class A
{
   abstract void callme();
   void callmetoo() {
      System.out.println("This is a concrete method.");
   }
}
class B extends A
{
   void callme() {
      System.out.println("B's implementation of callme.");
   }
}
   class Main {
   public static void main(String args[]) {
   B b = new B();
   b.callme();
   b.callmetoo();
   }
}
```

Output-

```
B's implementation of callme.
This is a concrete method.
```

2.

```
abstract class A
{
   abstract void callme();
   void callmetoo() {
      System.out.println("This is a concrete method.");
   }
}
class B extends A
{
   void callme() {
      System.out.println("B's implementation of callme.");
   }
}
   class Main {
   public static void main(String args[]) {
   B b = new B();
   b.callme();
   b.callmetoo();
   }
}
```

Output-

```
B's implementation of callme.
This is a concrete method.
```

3.

```
abstract class A
{
   abstract void callme();
}
class B extends A
{
   void callme() {
      System.out.println("B's implementation of callme.");
   }
}
class Main
{
   public static void main(String args[]) {
   B b = new B();
   b.callme();
   }
}
```

Output-

```
B's implementation of callme.
```

4.

```
abstract class A
{
   abstract void callme();
   void callmetoo() {
      System.out.println("This is a concrete method.");
   }
}
class B extends A
{
   void callme() {
      System.out.println("B's implementation of callme.");
   }
}
   class Main {
   public static void main(String args[]) {
   B b = new B();
   b.callmetoo();
   }
}
```

Output-

```
This is a concrete method.
```