```java
class A {}
class B extends A {}
class C extends A {}

// ...
public class Example{


A a = new A();
B b = new B();
C c = new C();

boolean isAAnInstanceOfA = a instanceof A; // true

boolean isBAnInstanceOfA = b instanceof A; // true
boolean isBAnInstanceOfB = b instanceof B; // true

boolean isCAnInstanceOfA = c instanceof A; // true
boolean isCAnInstanceOfC = c instanceof C; // true


// The following will fail to build with the message:
// error: incompatible types: C cannot be converted to B
// boolean isCAnInstanceOfB = c instanceof B;
    }


    class A {}

    class B extends A {
        int x;

        B(int x) {
            this.x = x;
        }
    }

    class C extends B { // Note that this is different to the example in `instanceof`!
        int y;

        C(int x, int y) {
            super(x);
            this.y = y;
        }
    }
    public class Example{
    // ...
    A a = new B(1);
    A a2 = new C(2, 3);

    // This will show the following error message:
    // error: cannot find symbol
    //   int error = a.x;
    //               ^
    // symbol:   variable x
    // location: variable a of type A
    // int error = a.x;

    // This works, and assigns 1 to x
```

```
60      B b = (B) a;
61      int x = b.x;
62
63      // We don't have to create a new variable, but then
64      // we need to wrap the cast object with parentheses
65      int cX = ((B) a2).x;
66      int cY = ((C) a2).y;
67      }
68
69
70      class A {
71          private int count;
72          private int x;
73
74          A(int x) {
75              this.x = x;
76              this.count = 0;
77          }
78
79          public int getX() {
80              this.count += 1;
81              return this.x;
82          }
83
84          public void setX(int x) {
85              this.count += 1;
86              this.x = x;
87          }
88
89          public int getAccessCount() {
90              return this.count;
91          }
92      }
93
94      // ...
95      public class Example{
96          public static void main(String args[]){
97      A a = new A(5);
98
99      // This causes an error:
100     // error: x has private access in A
101     //   int error = a.x;
102     //               ^_^
103     // int error = a.x;
104
105     // Let's access x a bunch
106     int currentValue = a.getX();
107     a.setX(a.getX() + 3); // Note that we access x twice here!
108     int newValue = a.getX();
109
110     int accessesToX = a.getAccessCount();
111     System.out.println(currentValue);
112     System.out.println(newValue);
113
114         }
115     }
```