Key point: *Common elementary programming errors often involve undeclared variables, uninitialized variables, integer overflow, unintended integer division, and round-off errors.*

**Common Error 1: Undeclared/Uninitialized Variable and Unused Variables**

Example 1: undeclared variable

```
double interestRate = 0.05;
double interest = interestrate * 45;   //Variable interestrate has never been declared.
                                       //Recall Java is case sensitive:
                                       //interestrate and interestRate are different
                                       //identifiers
```

Example 2: unused variable

```
double interestRate = 0.05;
double taxRate = 0.05;                 //Variable taxRate is never used and should be
                                       //discarded.
                                       //An IDE, like NetBeans, will notify the programmer
                                       //when a variable is not used.
double interest = interestRate * 45;
System.out.println("Interest is " + interest);
```

**Common Error 2: Integer Overflow**

Example 1: Unintended positive to negative sign reversal

```
int value = 2147483647 + 1;            //value will actually be -2147483648
```

```
    7  F  F  F  F  F  F  F   2147483647
+                        1            1
-------------------------------------------
    8  0  0  0  0  0  0  0

    7  F  F  F  F  F  F  F   1's complement
    8  0  0  0  0  0  0  0   2's complement
```

Example 2: Unintended negative to positive sign reversal

```
int value = -2147483648 - 1;           //value will actually be 2147483647
```

**Common Error 3: Round-off Errors**

Example 1: Inexact floating-point computation

```
System.out.println(1.0 – 0.1– 0.1– 0.1– 0.1– 0.1);
//Displays 0.5000000000000001, not 0.5, and
System.out.println(1.0 – 0.9);
//Displays 0.09999999999999998, not 0.1
```

Integers are stored precisely – floating-point values are approximate.

### Common Error 4: Unintended Integer Division

Example 1: A floating-point result is desired but an integer result is computed

```
//incorrect
int number1 = 1;
int number2 = 2;
double average = (number1 + number2) / 2;     //average = 1.0

//correct
int number1 = 1;
int number2 = 2;
double average = (number1 + number2) / 2.0;      //average = 1.5
```

### Common Pitfall 1: Redundant Input Objects

Example 1: Two scanners

```
//Bad Code
Scanner input = new Scanner(System.in);
System.out.print("Enter an integer: ");
int v1 = input.nextInt();

Scanner input = new Scanner(System.in);           //A new scanner is created and
                                                  //assigned to variable input
                                                  //making the original scanner
                                                  //inaccessible
System.out.print("Enter an double value: ");
double v2 = input.nextDouble();


//Good Code
Scanner input = new Scanner(System.in);
System.out.print("Enter an integer: ");
int v1 = input.nextInt();
System.out.print("Enter an double value: ");
double v2 = input.nextDouble();
```