Chapter 2 Primitive Data Types and Operations

1. Valid identifiers: applet, Applet, $4, apps
   Invalid identifiers: a++, --a, 4#R, #44

2. Keywords:
   class, public, int

3. double miles = 100;
   final double MILE_TO_KILOMETER = 1.609;
   double kilometer = MILE_TO_KILOMETER * miles;
   System.out.println(kilometer);

   The value of kilometer is 160.9.

4. There are three benefits of using constants: (1) you don't have to repeatedly type the same value; (2) the value can be changed in a single location, if necessary; (3) the program is easy to read.

   final int SIZE = 20;

5.

a = 46 / 9; => a = 5

a = 46 % 9 + 4 * 4 - 2; => a = 1 + 16 – 2 = 15

a = 45 + 43 % 5 * (23 * 3 % 2); => a = 45 + 3 * (1) = 48

a %= 3 / a + 3; => a %= 3 + 3; a % = 6 => a = a % 6 = 1;

d = 4 + d * d + 4; => 4 + 1.0 + 4 = 9.0

d += 1.5 * 3 + (++a); => d += 4.5 + 2; d += 6.5; => d = 7.5

d -= 1.5 * 3 + a++; => d -= 4.5 + 1; => d = 1 – 5.5 = -4.5

6.

2

2

-4

-4

0

1

7. For byte, from -128 to 127, inclusive.
   For short, from -32768 to 32767, inclusive.
   For int, from -2147483648 to 2147483647, inclusive.
   For long, from -9223372036854775808 to 9223372036854775807.

For float, the smallest positive float is 1.40129846432481707e-45 and the largest float is 3.40282346638528860e+38.
For double, the smallest positive double is 4.94065645841246544e-324 and the largest double is 1.79769313486231570e+308d.

8. 25/4= 6. If you want the quotient to be a floating-point number, rewrite it as 25.0/4.0.

9. Yes, the statements are correct. The printout is

the output for 25 / 4 is 6;
the output for 25 / 4.0 is 6.25;

10. 4.0 / (3.0 * (r + 34)) – 9 * (a + b * c) + (3.0 + d * (2 + a)) / (a + b * d)

11. b and c are true.

12. All.

13.
Line 2: Missing static for the main method.
Line 2: string should be String.
Line 3: i is defined but not initialized before it is used in Line 5.
Line 4: k is an int, cannot assign a double value to k.
Lines 7-8: The string cannot be broken into two lines.

14. Yes. Different types of numeric values can be used in the same computation through numeric conversions referred to as *casting*.

15. The fractional part is truncated. Casting does not change the variable being cast.

16.
f is 12.5
i is 12

17.

```
System.out.println((int)'1');
System.out.println((int)'A');
System.out.println((int)'B');
System.out.println((int)'a');
System.out.println((int)'b');

System.out.println((char)40);
```

```
System.out.println((char)59);
System.out.println((char)79);
System.out.println((char)85);
System.out.println((char)90);

System.out.println((char)0X40);
System.out.println((char)0X5A);
System.out.println((char)0X71);
System.out.println((char)0X72);
System.out.println((char)0X7A);
```

18. '\u345dE' is wrong. It must have exactly four hex numbers.

19. '\\' and '\”'

20.

i becomes 49, since the ASCII code of '1' is 49;
j become 99 since (int)'1' is 49 and (int)'2' is 50;
k becomes 97 since the ASCII code of 'a' is 97;
c becomes character 'z' since (int) 'z' is 90;

21.

```
char c = 'A';
i = (int)c; // i becomes 65

boolean b = true;
i = (int)b; // Not allowed

float f = 1000.34f;
int i = (int)f; // i becomes 1000

double d = 1000.34;
int i = (int)d; // i becomes 1000

int i = 97;
char c = (char)i; // c becomes 'a'
```

22.

```
System.out.println("1" + 1);  => 11
System.out.println('1' + 1);  => 50  (since the Unicode for 1 is 49
System.out.println("1" + 1 + 1);  => 111
System.out.println("1" + (1 + 1)); => 12
System.out.println('1' + 1 + 1); => 51
```

23.

1 + "Welcome " + 1 + 1 is 1Welcome 11.

1 + "Welcome " + (1 + 1) is 1Welcome 2.

1 + "Welcome " + ('\u0001' + 1) is 1Welcome 2

1 + "Welcome " + 'a' + 1 is 1Welcome a1

24. Use Double.parseDouble(string) to convert a decimal string into a double value.
Use Integer.parseInt(string) to convert an integer string into an int value.

25. long totalMills = System.currentTimeMillis() returns the milliseconds since Jan 1, 1970. totalMills % (1000 * 60 * 60) returns the current minute.

26. Use // to denote a comment line, and use /* paragraph */ to denote a comment paragraph.

27. Class names: Capitalize the first letter in each name.
Variables and method names: Lowercase the first word, capitalize the first letter in all subsequent words.
Constants: Capitalize all letters.

28.

```
public class Test {
  /** Main method */
  public static void main(String[] args) {
    // Print a line
    System.out.println("2 % 3 = " + 2 % 3);
  }
}
```

29. Compilation errors are detected by compilers. Runtime errors occur during execution of the program. Logic errors results in incorrect results.