なぜ

```
printf("平均値 = %10.4f¥n", (x+y+z)/3 );
```

ではうまく動かないのか？

まず，printf では出力の書式として "平均値 = %10.4f¥n" と指定されていますので，答えを「平均値 = ○○○（最後に改行）」というかたちで表示しようとします．○○○の部分は %10.4f という指定から，全体で 10 桁分の幅をとり，小数点以下は 4 桁までの実数を表示させようとします．実数のデータ型は double 型となります．

表示を行うには，○○○の部分に入る具体的な数値を求める必要があります．これに該当するのが (x+y+z)/3 の計算結果というわけです．

次に，この数式の計算を考えてみます．例として，x=1,y=2,z=4 だとしましょう．すると

```
(x+y+z)  /   3
    ↓
(1+2+4)  /   3
    ↓
   7     /   3
```

という流れになり，7/3 を計算することになります．

ここで，x, y, z はいずれも int 型でしたので，上の過程で得られた 7 も int 型です．
一方，3 も（小数点は付いていないので）int 型として解釈されます．

よって，int 型と int 型の割り算（整数と整数の割り算）という扱いになり，答えも int 型（整数）として求められます．したがって，この計算の答えは 2.333333 ではなく，小数点以下が切り捨てられた整数 2 になるのです．

こうして，もともとの printf は

```
printf("平均値 = %10.4f¥n", (x+y+z)/3 );
                                  ↓
printf("平均値 = %10.4f¥n",     2     );
```

という解釈になります．
つまり，printf は 2 という整数を「全体で 10 桁分の幅をとり，小数点以下は 4 桁までの実数」として表示させようとするので，2.0000 という誤った答えが得られてしまうのです．

このように，**表示部分のみを double 型に合わせるだけではダメ**なのです．

（裏面へ続く）

計算式の中で，**割り算を行う前に** int 型から double 型へ変換しておかないと小数点以下が捨てられてしまいます．
そのような変換を行うのが，前回学習したキャストなのです．

具体的には `(x+y+z)` の前に `(double)` と書きます．これにより計算式の扱いは

```
(double)(x+y+z)  /  3
        ↓
(double)(1+2+4)  /  3
        ↓
   (double)7     /  3
        ↓
     7.000000   /  3
```

という流れに変わります．
よって，`7.000000/3` を計算することになり，今度はうまく `2.333333` となるのです．

ポイントは，**割り算を行う前に分子を double 型に変換**することです．これを理解してもらえると次の例も間違いであることが分かるでしょう．以下の誤例では，括弧が式全体についているので，double への変換よりも割り算が先に計算されることになり，上の間違いと同様にうまくいきません．

```
誤：(double)((x+y+z)/3)

正：(double)(x+y+z)/3
```