基礎プログラミング演習 II 教材 (#5)

■ 動作時の異常・エラー

前回の課題 3. は計算結果の異常、あるいはエラー（処理不能）を調べるものでした。代表的なものを紹介します。

□ ゼロによる除算

ゼロによる除算はエラーを生じさせます。
この場合、処理は途中で止まっており、最後の exit() まで到達していません。
「Floating Exception」の表示はエラーメッセージであり、printf 関数が出したものではありません。

```
$ ./answer5
Number 1: 100
Number 2: 0
和 = 100
積 = 0
Floating exception
$
```

□ 桁あふれ

入力に巨大な数を指定した場合、正しい結果が得られません。
ところでどの段階で異常が生じたのでしょう。
計算処理がおかしいのでしょうか？
それ以前の問題でしょうか？

```
$ ./answer5
Number 1: 888888888888888888888888888
Number 2: 88888888888888888888888
和 = -2
積 = 1
商 = 1
$
```

□ プログラマの責任

プログラマは動作途中にエラーが生じないよう、よく考慮してプログラムを作成しなければなりません。また不適切な処理が常にエラーを出してくれるとは限りません。
つまり問題が生じない範囲で処理が進行し、異常な動作が起きないように注意しながらプログラムを書く必要があるのです。

プログラムがどのように動くのか、その内実をよくよく把握しなければなりません。
しかし課題をただ動作させることだけに集中して作業してはいつまで経っても何も見えません。
このギャップの意味を考えなければなりません。

例えば上のプログラムでは 6 桁程度の数値を入力した場合はまた違った状況が発生します。何が起きているのか、といったことは後の「データ型」の単元で説明します。
（興味のある受講生は教科書 p.21 の 2.2.1 を参照）

宿題
次回はループ（繰り返し処理）です。
次回教材範囲と教科書 p. 75〜79 を予習するように。