入出力処理

三池　克明

関数 printf()と新たに学ぶ関数 scanf()を使ってデータの入出力処理を解説します。
特に scanf()は対話式プログラム（ユーザーに操作を促すプログラム）を作るうえで重要です。

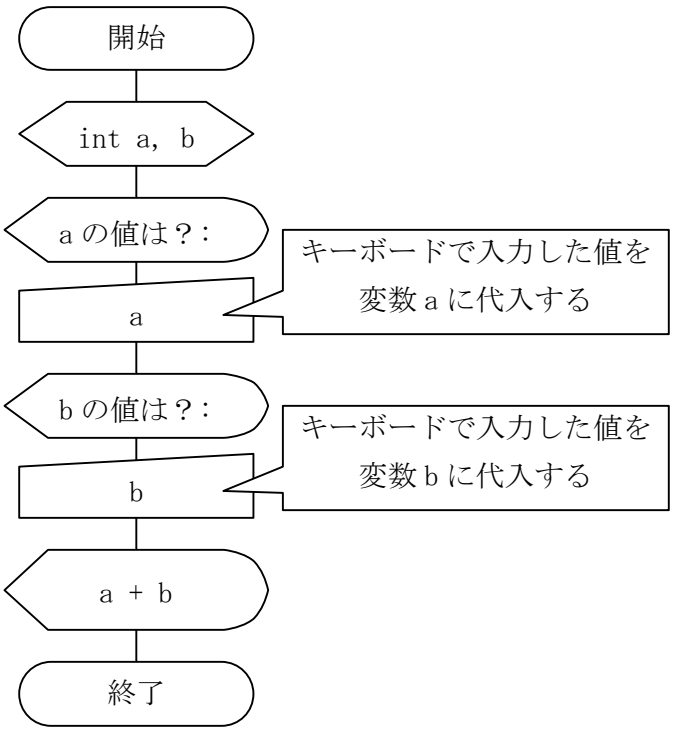
—目　次—

1.　関数 scanf()

関数 printf()は変数の値を画面に表示しますが、それに対し関数 scanf()はキーボードで入力した値を変数に代入します。
この関数を活用することで対話式（ユーザーの操作に応じて処理を行う）プログラムを作ることができるようになります。

1.1.　２整数の和を求める

以下のフローチャートどおりに処理するプログラムを作成します。



scanf1.cpp

```
 1:  /* ２整数の加算*/
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          int     a, b;
 7:
 8:          printf("a の値は？: ");
 9:          scanf("%d", &a);
10:          printf("b の値は？: ");
11:          scanf("%d", &b);
12:
13:          printf("a + b = %d + %d = %d¥n", a, b, a + b);
14:  }
```

変数名の手前の
＆に注意

変数名の手前の
＆に注意

実行例

```
a の値は？: 35
b の値は？: 51
a + b = 35 + 51 = 86
```

実行時に
キーボードで入力

このように実行時に入力した値を使って計算をします。

なお、今回使用した関数 scanf()の構文は以下のとおりです。

構文：

scanf(*データを取り出す書式, &変数名*)

ヘッダファイル：

stdio.h

「データを取り出す書式」は%d などのように関数 printf()と同じものを使用できます。

ただし、**変数名には手前に＆が付くことに注意してください**。

1.2. 入力した文字を得る

今度は文字を入力するプログラムです。

scanf2.cpp

```
 1:  /* 文字を入力する */
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          char    moji;
 7:
 8:          printf("何か半角文字を入力してください : ");
 9:          scanf("%c", &moji);
10:
11:          printf("あなたが入力した文字は[%c]ですね。¥n", moji);
12:          printf("また、[%c]の文字コードは[%d](10進数)です。¥n", moji, moji);
13:  }
```

実行例

```
何か半角文字を入力してください : K
あなたが入力した文字は[K]ですね。
また、[K]の文字コードは[75](10進数)です。
```

―プログラム内部でのデータの扱い―

scanf2.cpp で入力した文字を整数として表示する部分があります。

このようなことができるのはコンピュータがあらゆるデータを数値として扱っているためです。

よって 'A'+ 1 といった計算も可能です。

（ちなみに 'A' + 1 = 'B' となります）

2.　入出力処理と計算

ここでは関数 printf()と関数 scanf()を使った計算プログラムを例に挙げ解説します。

2.1.　２整数の商を求める

はじめに割り算のプログラムを作ってみましょう。

scanf3.cpp

```
 1:  /* ２整数の除算*/
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          int     a, b;
 7:          double  c;
 8:
 9:          printf("a の値は？: ");
10:          scanf("%d", &a);
11:          printf("b の値は？: ");
12:          scanf("%d", &b);
13:
14:          c = a / b;
15:
16:          printf("a ÷ b = %d ÷ %d = %f¥n", a, b, c);
17:  }
```

実行例

```
a の値は？: 10
b の値は？: 3
a ÷ b = 10 ÷ 3 = 3.000000
```

おや？　おかしいですね。
割り算の結果を代入する変数 c は double 型なので小数も扱えるはずです。

では、どうしてこのようなことになったのでしょう。

C 言語では int 型同士の演算結果を int 型にしてしまいます。
よって、

```
c = 10 / 3;
c =    3    ←int 型と int 型の演算結果は int 型になる
```

よって変数 c に代入されるのは int 型の 3 となり、変数 c の値は 3.0 となるわけです。

そこでキャスト演算子を使ったのが以下の「scanf4.cpp」です。
ソースコードを入力し、コンパイル、実行して結果を確かめてみましょう。

scanf4.cpp

```
 1:  /* ２整数の除算-キャスト演算子を使う*/
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          int     a, b;
 7:          double  c;
 8:
 9:          printf("a の値は？: ");
10:          scanf("%d", &a);
11:          printf("b の値は？: ");
12:          scanf("%d", &b);
13:
14:          c = (double)a / b;
15:
16:          printf("a ÷ b = %d ÷ %d = %f¥n", a, b, c);
17:  }
```

実行例

```
a の値は？： 10
b の値は？： 3
a ÷ b = 10 ÷ 3 = 3.333333
```

上手くいきましたね。
それでは重要な部分を説明しましょう。

```
13:
14:             c = (double)a / b;
15:
```

ここでキャスト演算子を使って変数 a を double 型にキャスト（型変換）しています。
キャスト演算の書式は、

(型名)変数名

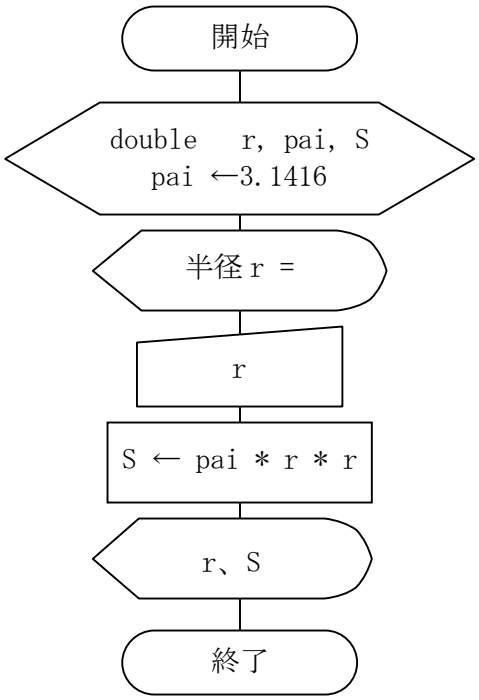
と理解すればほぼ問題ありません（詳しい解説は本書では割愛します）。
また double 型と int 型を含む式を演算するとき、その結果は double 型になります。
よって、

```
c = (double)10 / 3;     ←キャスト演算される
c =     10.0   / 3;     ←double 型になる
c = 3.33333333          ←doule 型と int 型の演算結果は double 型になる
```

となり、変数 c には double 型の 3.3333333 となります。

2.2. 円の面積を求める

半径の値を入力させ、その円の面積を求めるプログラムを作りましょう。
そのフローチャートは以下のとおりです。



そしてソースプログラムは以下のとおりです。
実行結果を確かめてみましょう。

en-mensekil.cpp

```
1:  /* 円の面積を求める */
2:  #include <stdio.h>
3:
4:  main()
5:  {
6:          double  S, r, pai = 3.1416;
7:
8:          printf("半径 r[cm] = ");
```

```
 9:          scanf("%lf", &r);
10:
11:          S = pai * r * r;
12:          printf("半径%f[cm]の円の面積Sは%f[cm^2]です。¥n", r, S);
13:  }
```

%f ではなく%l(エルエフ) f にする

実行例

```
半径 r[cm] = 33.8
半径 33.800000[cm]の円の面積 S は 3589.089504[cm^2]です。
```

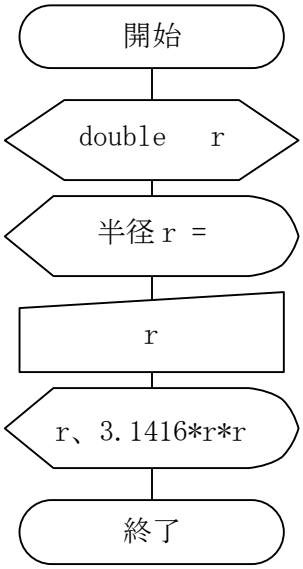
ここで9行目の関数 scanf()をみてみましょう。

```
 8:          printf("半径 r[cm] = ");
 9:          scanf("%lf", &r);
10:
```

%f ではなく%l(エルエフ) f にする

scanf()で double 型の値を入力させるときは、**"%f"ではなく"%l(エルエフ) f"と記述しましょう。**

次に en-menseki1.cpp を改良したプログラムです。



そしてソースプログラムです。
en-menseki1.cpp と比べて、

- 宣言する変数が少ない
- 関数 printf()の中に計算式を記述している

ことが分かります。

en-menseki2.cpp

```
 1:  /* 円の面積を求める 2 */
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          double   r;
 7:
 8:          printf("半径 r[cm] = ");
 9:          scanf("%lf", &r);
10:
11:          printf("半径%f[cm]の円の面積 S は%f[cm^2]です。¥n", r, 3.1416 * r * r);
12:  }
```

実行例

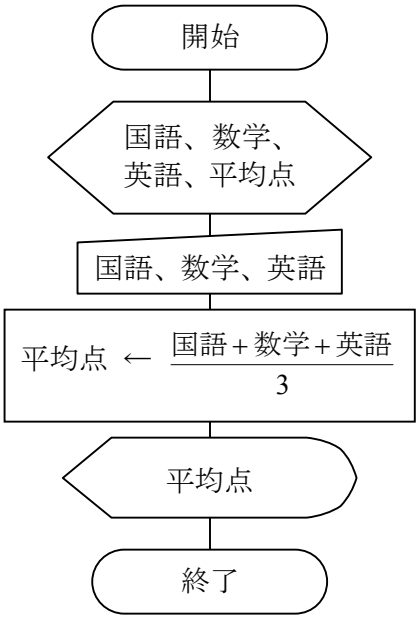
```
半径 r[cm] = 33.8
半径 33.800000[cm]の円の面積 S は 3589.089504[cm^2]です。
```

ソースプログラムの内容は若干異なりますが、実行するうえでは何ら変らないことが分かります。

2.3. 三科目（国語、数学、英語）の平均点を求める

国語、数学、英語の平均点を求めるプログラムを作りましょう。
内容がやや難しいので以下の表も作りました。
フローチャート、ソースプログラムと見比べれば理解しやすくなると思います。

フローチャートでの変数名	ソースプログラムでの変数名	型
国語	kokugo	int
数学	sugaku	int
英語	eigo	int
平均点	average	double



heikinten1.cpp

```
1:  /* 国語、数学、英語の平均点を求める */
2:  #include <stdio.h>
3:
4:  main()
5:  {
```

```
 6:          int     kokugo, sugaku, eigo;
 7:          double  average;
 8:
 9:          printf("国語の点数 = ");
10:          scanf("%d", &kokugo);
11:          printf("数学の点数 = ");
12:          scanf("%d", &sugaku);
13:          printf("英語の点数 = ");
14:          scanf("%d", &eigo);
15:
16:          average = (kokugo + sugaku + eigo) / 3.0;
17:
18:          printf("平均点は%f です¥n", average);
19:  }
```

実行例

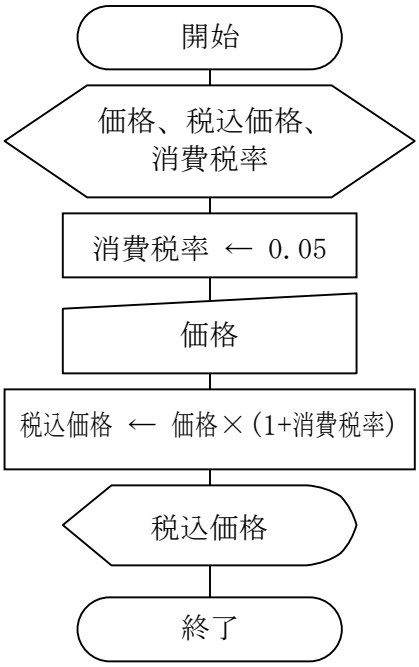
```
国語の点数 = 80
数学の点数 = 83
英語の点数 = 79
平均点は 80.666667 です
```

きちんと平均点が算出されているのが分かります。

2.4. 消費税込みの価格を計算する

消費税込価格の計算をするプログラムを作ってみましょう。
こちらもやや難しいので表も作りました。

フローチャートでの変数名	ソースプログラムでの変数名	型
価格	kakaku	int
消費税率	syohizei	double
税込価格	zeikomi	int



syohizei.cpp

```
 1:  /* 消費税込価格を算出する */
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          int     kakaku, zeikomi;
 7:          double  syohizei;
 8:
 9:          syohizei = 0.05;
10:
11:          printf("価格を入力してください : ");
12:          scanf("%d", &kakaku);
13:
14:          zeikomi = kakaku * (1 + syohizei);
15:
16:          printf("税込価格は¥¥%d です。¥n", zeikomi);
17:  }
```

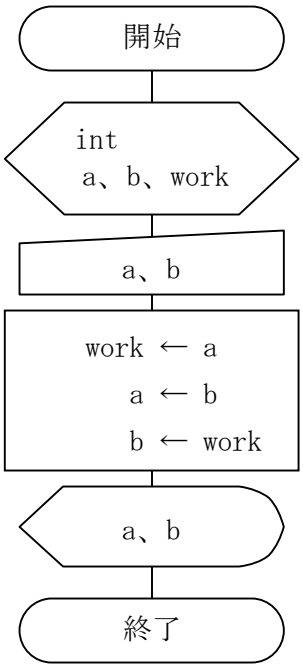
実行例

```
価格を入力してください : 1200
税込価格は¥1260 です。
```

税込価格がきちんと計算されているのが分かります。

2.5. ２変数の値を入れ替える

二つの変数の値を入れ替えるプログラムを作ってみましょう。
意外な発見があるはずです。



swap1.cpp

```
 1:  /* ２変数の値を入れ替える */
 2:  #include <stdio.h>
 3:
 4:  main()
 5:  {
 6:          int     a, b, work;
 7:
 8:          printf("a = ");
 9:          scanf("%d", &a);
10:          printf("b = ");
11:          scanf("%d", &b);
12:
13:          work = a;
```

```
14:             a = b;
15:             b = work;
16:
17:             printf("入れ替えた結果¥n");
18:             printf("a = %d, b = %d¥n", a, b);
19:     }
```

実行例

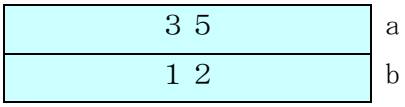
```
a = 35
b = 12
入れ替えた結果
a = 12, b = 35
```

変数の入れ替えというと大抵の場合、

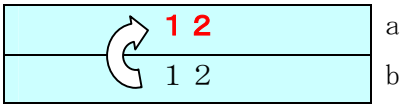
```
a = b;
b = a;
```

と考えてしまいがちですが、これは間違いです。
なぜ間違いなのかは、順番に処理をたどっていけば分かります。

①a と b の状態



②a = b;
変数 b の値が変数 a に代入されます。

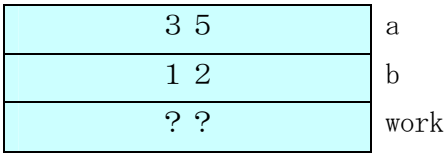


この時点で元々の変数 a の値「35」が失われたため、入れ替えに失敗していることが分かります。

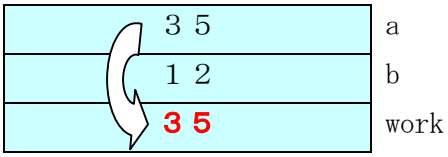
そこで swap1.cpp では、以下のように変数をもう一つ用意して処理をします。

```
13:             work = a;
14:             a = b;
15:             b = work;
```

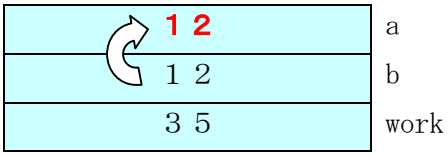
①a と b の状態



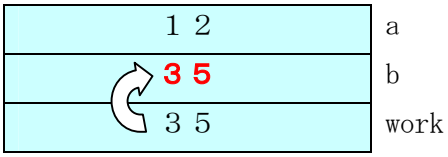
②13 行目「work = a;」



③14 行目「a = b;」



④15 行目「b = work;」



このように値の入れ替えができているのがわかります。

3. 演習問題

3-1. 直方体の各辺の長さ a[cm]、b[cm]、c[cm]を入力し、その直方体の表面積 S[cm^2]と体積 V[cm^3]を算出し表示するプログラム「ensyu3-1.cpp」を作りなさい。

実行例

```
直方体の３辺の長さをａ、ｂ、ｃを入力してください
a[cm] = 3.5
b[cm] = 4
c[cm] = 5.2
表面積 S は 106.000000[cm^2]、体積 V は 72.800000[cm^3]です。
```

ヒント：

まずは表面積 S と体積 V の式を組み立てましょう。

3-2. 一次方程式 $Ax+B=0$ がある。A と B を入力させ方程式を表示するプログラム「ensyu3-2.cpp」を作りなさい。なお A、B は整数とする。

実行例

```
x の一次方程式 Ax+B=0 の A と B を入力してください。
ただし A、B は共に整数とします。
A = 12
B = -5
方程式は 12x-5=0 ですね。
```

ヒント：

printf()関数で符号付で表示させる方法を学びましたね。

3-3.　「ensyu3-2.cpp」を発展させ、x の解を算出するプログラム「ensyu3-3.cpp」を作りなさい。

実行例

```
x の一次方程式 Ax+B=0 の A と B を入力してください。
ただし A、B は共に整数とします。
A = -4
B = -7
方程式 -4x-7=0 の解は
-1.750000 です
```

ヒント：

A、B は整数ですが x は実数です。よって解の計算にはキャスト演算子が必要です。