プログラミング演習 第3回
制御文 (if else, for)

目 次

1 if else による条件分岐 **1**

1.1 数の大小比較 1

1.2 数学ライブラリの使い方 2

2 演習 **4**

3 関係演算子と論理演算子 **5**

4 付録: 二次方程式の解法 **5**

今日は, if else, for 文を使った制御文について学びます. 内容として高校で取り上げるであろうものです.

次回は優しい文字列操作と while 文について触れるつもりです.

1 if else による条件分岐

1.1 数の大小比較

前回使用した for 文は，プログラムを繰り返し実行させる制御という機能を持った文です．繰り返しと並んで重要な制御のための文は if 文です．if 文は条件により，処理の分岐を行います．条件分岐のコマンドにはこの他に switch 文があります．

if 文と for 文があれば，殆どのプログラムを作ることが出来ます．次回はさらに while 文も扱います．while 以外の繰り返しを行うための命令は，do while があります．

do while と switch の機能については自習して下さい．

if 文を使ったプログラムを作成しましょう (ファイル名 3-１.c)．

```
/* ２つの整数を比較する */
#include <stdio.h>

main()
{
  int      m, n;

  printf("２つの整数を空白で区切って入力して下さい >> ");

  scanf("%d %d", &m, &n);

  if (m > n) {
    printf("%d は %d より大きい\n", m, n);
  } else if (m == n) { /* 等しいかどうかの比較は == を使用 */
    printf("%d は %d と等しい\n", m, n);
  } else {
    printf("%d は %d より小さい\n", m, n);
  }
```

}

前回使用した関数 scanf は書式付き入力関数と呼ばれるもので, 標準入力 (今の場合キーボードからの入力) から入力されたデータを指定した書式に変換して変数に格納します. プログラム 3-1.c にある例

```
scanf("%d %d", &m, &n)
```

は, m に最初の整数 (d) の値が代入され、n に次の整数 (d) の値が代入されるということです. (三つ以上も同じ) 注意として, 変数の前に&(アンパサンド) を忘れずに.

if 文の使い方は次の通りです．
条件が真のときのみ，ある処理を行いたいならば，

```
if（条件式）{
  条件式が真ならここを実行
}
```

条件式が偽のとき別の処理をしたいならば，

```
if（条件式）{
  条件式が真ならここを実行
} else {
  条件式が偽ならここを実行
}
```

条件式が偽のとき，別の条件を調べたいなら

```
if（条件式 1）{
  条件式 1 が真ならここを実行
} else if（条件式 2 ){
  条件式 1 が偽で，かつ，条件式 2 が真ならここを実行
}
```

1.2 数学ライブラリの使い方

数学ライブラリを使うソースファイルを if 文を用いて書きます. 先週のレポート問題 1 を関数 pow(double x, double y) を用いて書きましょう (ファイル名 3-2.c). 但し, 求める指数の値は任意の実数とします.

この例では, 数学関係ライブラリを使うのでそこで扱う関数を定義するために最初に math.h というファイルを include しています.

```
/* 2^n を求めるプログラム */
#include <stdio.h>
#include<math.h>

main()
{

  double n;                        /* 指数 */
  double power;                      /* 求める解*/


  printf("求める指数 n を入力して下さい\n");
  scanf(  "%lf", &n);      /* n に値を代入 */

  power = pow(2.0,n);
  printf("求める値は%f です\n", power);
  }
```

数学関数ライブラリの使い方

上のプログラムは, 通常の cc コマンド ではコンパイルできません.

```
cc1.cc % cc 3-2.c
```

を実行して, 出て来るメッセージを確認してください. 私の環境では, 次のようになりました.

```
%cc 3-2.c
Undefined                          first referenced
 symbol                                in file
pow                                   3-2.o
ld: fatal: Symbol referencing errors. No output written to a.out
```

エラーメッセージは, 上のような表形式でてます. 1-2 行目は, 左側に未定義シンボルと書いてあって, 右側に最初に参照されたファイルと書いてあります. 単に翻訳しただけですが, この程度の翻訳はできるのが, 大学

生の常識です[1] . 3 行目は, 上の物の内容で, 未定義シンボルが pow, 参照されたファイルが, 3-2.o という意味です. 4 行目の先頭は, このメッセージを出したコマンドで ld (link editor リンカ) でその後に書いてあるような事を言っています. (さて, なんて書いてあるのでしょう ?)

これは, ld がプログラム a.out を作り上げる際に pow という関数がどこにも無かったので, 最後まで実行できなかったため出て来たエラーメッセージです. UNIX の cc(ld) は, デフォルトでは標準 C ライブラリ /usr/lib/libc.a(/usr/lib/libc.so) にある関数しか, ライブラリをリンクしません. (自分でチェックしよう) pow は数学関数ライブラリにありますから, このような事が起こります. 標準ライブラリ以外のライブラリにある関数をリンクするには, オプション -l を cc に加えてコンパイルします. ちなみに, 数学関数ライブラリは, /usr/lib/libm.a(/usr/lib/libm.so) ですが, これをリンクするには次のようにコマンドを打ちます.

```
cc1.cc % cc 4-1.c -lm
```

-l の後の文字 m の意味は上のファイルから類推できるでしょう[2] . 他のライブラリをリンクする時も, 同様の規則で -l の後のアルファベットを指定します.

scanf についての注意

ソースファイル 3-2.c の 11 行目にある`scanf` は, double 型の変数に値を代入しますので`%lf` としなければなりません. ここが, 関数`printf` と多少違います.

その他の注意

関数 pow の引数 (変数) は double 型であるので, 代入する整数は 2.0 のように double 型にしたほうがよろしいです. そのようにしない場合は, たとえコンパイルが通っても結果がおかしくなることがあります.

2 演習

演習時間に課題を与えますので, 授業内に作成し, メールで提出して下さい. 課題名は (Subject:の書く名前) creport.3-0 です.

授業内にできない方も提出すれば評価します (但し, 評価は多少下がる).

[1] ほとんどの大学生が常識が無いと言う, 悲しい現実がありますが

[2] もちろん m は mathematical function の m です

3　関係演算子と論理演算子

for 文，if 文などに出てきた大小比較の記号 <，> は正しくは関係演算子と呼ばれます．関係演算子はこの他に次のものがあります．

<= 等しいか，より小さい．≦ と同じ意味である．

>= 等しいか，より大きい．≧ と同じ意味である．

!= 等しくない．≠ と同じ意味である．

同じく制御文によく使われる論理演算子と呼ばれるものがあります．論理演算子は次の 3 つです．

論理積 記号&&（`A && B` は A, B が共に真のとき真となる）．

論理和 記号||（`A || B` は A, B どちらかが真のとき真となる）．

否定 記号!（`!A` は A が真のとき偽，A が偽のとき真となる）．

これらを組合せて条件式を作ることができます．例えば，下の式は，いつ真になるか考えましょう．

```
5<a && a<10
```

上記の演算の結果は整数に値を持ちます．例えば，`a = 5 < 3`, `a = 3 < 5`, `a = 1 || 0`, `a = 1 && 0`, `a = !1`, `a = !3`, `a = !0`, などとすると `a` に整数が代入されます．どんな値が代入されているか調べてみるとよろしいです．論理積，論理和という言葉の意味がわかるでしょう．他の例も調べてみよう．

4　付録: 二次方程式の解法

二次方程式の解を求めるプログラムを if 文を用いて書きます. すぐに思いつくのは, 解の公式を用いた次のプログラムです.

この例でも, 数学関係ライブラリを使うのでそこで使う関数を定義するために最初に math.h というファイルを include しています. 使う関数は, sqrt で平方根をとる関数です.

```
/* 二次方程式の実数解の解法１ */
#include <stdio.h>
#include<math.h>

main()
{

  double a, b, c;                        /* 二次方程式の係数 */
  double x1, x2, D;                      /* 求める解と判別式  */

  printf("係数を a b c 入力して下さい\n");
  scanf(  "%lf %lf %lf", &a, &b, &c);     /* a, b, c に値を代
入 */
  if(a==0.0)
    {
  if(b==0.0)
    {  printf("係数がおかしい\n");
     }
  else
    { x1 = -c/b;
        printf("解は%f です\n", x1);
    }
}
else
    {
  D = b*b - 4.0*a*c;                          /* 判別式Dの宣言
*/
     if(D >=  0.0)
       {
```

```
        x1 = (-b + sqrt(D))/(2.0*a);
        x2 = (-b - sqrt(D))/(2.0*a);
        if(D==0.0)
          {
             printf("解は重解%f です\n",x1);
           }
        else
        {
             printf("解は%f %f です\n", x1,x2);
           }
      }
    else
      {
         printf("解は虚数解です\n");
       }
    }
}
```

実は解法 1 のプログラムは良いプログラムでありません桁落ちという誤差により, 真の解からのずれが大きい場合が起こります.

そこで解と係数の関係を用いたプログラムを一般に用います.

問題

入力された係数 b が 0 以上であるとき, 解の公式から x2 を求め, x1 を解と係数の関係から求める. 逆に, b が 0 より小さいとき, 解の公式から x1 を求め, x2 を解と係数から求めるプログラムを作成せよ.