関数 (追加説明)

山本昌志*

2006 年 12 月 1 日

概 要

先週で関数の学習を完了させるつもりであったが，理解度が不足している学生が見受けられたので，さらに練習を重ねる．

1 関数の作り方

1.1 三角形の面積の計算

1.1.1 プログラム作成方法

ヘロン公式

$$s = \frac{a+b+c}{2} \tag{1}$$

$$S = \sqrt{s(s-a)(s-b)(s-c)} \tag{2}$$

を使って三角形の面積 S を計算するプログラムを作成する．ここで，(a, b, c) は三角形の三辺の長さである．三角形の面積を計算するためには，この三辺の長さの情報で必要十分である．ここで，スモールエス s が負になると,「三角形の一辺は，他の二辺の和より短く，他の二辺の差よりも長い」という条件を満たさなくなり三角形が成立しない—ことにも注意してプログラムを作成しなくてはならない．

それでは，面積を計算する関数を考えよう．次のような関数が妥当だろう．

- 仮引数は辺の長さ `a` と `b`，`c` とする．もちろん，これらの仮引数の型は倍精度実数 (`double`) とすべき．
- 戻り値は，ヘロンの公式を使って計算した面積とする．
- もし，仮引数で渡された三辺では三角形ができなかった場合，戻り値を`-999` とする．三角形ができた場合の面積は必ず正である．もし戻り値が負の値となった場合，呼び出し元は渡した実引数が不適で，三角形が成立しなかったことが分かる．-999 のようなヘンテコリンな数字を戻り値とするのはプログラミングの定石である．

リスト 1 に面積を計算する関数を使った例を示す．ここで，三角形ができない場合のメイン関数の処理について注意が必要である．行目の `if(menseki < -990)` としていることを説明する．普通に考えれば，`if(menseki == -999)` としたいところではあるが，これは絶対にダメである．変数 `menseki` の型は倍精度

*独立行政法人　秋田工業高等専門学校　電気情報工学科

実数であるため，丁度-999.00000000000000となるとは限らないからである．倍精度実数には必ず誤差があるので，-999.00000000000001のように誤差がある可能性がある．このような理由から，倍精度実数を等価演算子==で比較することは厳に慎むべきである．これは分かりにくいバグの原因となる．一方，整数の場合は，誤差がないので等価演算子での比較で問題を起こすことはない．

リスト 1: 三角形の面積を計算するプログラム．

```
1  #include <stdio.h>
2  #include <math.h>
3
4  double helon(double a, double b, double c);  //プロトタイプ宣言
5
6  //=============================================================
7  // メイン関数
8  //=============================================================
9  int main(void)
10 {
11   double hen1, hen2, hen3;
12   double menseki;
13
14   printf("辺1の長さ?\t");
15   scanf("%lf",&hen1);
16   printf("辺2の長さ?\t");
17   scanf("%lf",&hen2);
18   printf("辺3の長さ?\t");
19   scanf("%lf",&hen3);
20
21   menseki = helon(hen1, hen2, hen3);
22
23   if(menseki < -990){
24     printf("入力した辺では，三角形はできません!!!!!\n");
25   }else{
26     printf("面積は，%fです．\n", menseki);
27   }
28
29   return 0;
30 }
31
32 //=============================================================
33 // ユーザー定義関数
34 //=============================================================
35 double helon(double a, double b, double c)
36 {
37   double s, S, test;
38
39   s=(a+b+c)/2;
40   test=s*(s-a)*(s-b)*(s-c);
41
42   if(test <= 0){
43     S = -999.0;
44   }else{
45     S = sqrt(test);
46   }
47
48   return S;
49 }
```

実行結果

```
辺 1 の長さ?       5.6
辺 2 の長さ?       8.3
辺 3 の長さ?       6.8
面積は，18.915076 です.
```

1.1.2 実行の流れと変数

1. プログラムの実行の準備[1]．すべて変数はローカルの自動変数なので，この時点で変数は無い．

2. メイン関数の実行の開始．メイン関数内にあるローカルの自動変数が生成される．生成される変数は初期化されていないので，値は不定である．

   ```
   メイン関数    hen1:?   hen2:?   hen3:?   menseki:?
   ```

3. キーボードからデータの読み取り完了 (19 行目実行直後).

   ```
   メイン関数    hen1:5.6   hen2:8.3   hen3:6.8   menseki:?
   ```

4. 関数 helon の呼び出し (21 行目実行直後)．このとき関数 helon の変数が作成される．そして，実引数の値が仮引数にコピーされる．

   ```
   メイン関数    hen1:5.6   hen2:8.3   hen3:6.8   menseki:?
   関数 helon    a:5.6      b:8.3      c:6.8      s:10.35       S:?       test:?
   ```

5. s と test の計算 (41 行目実行直後).

   ```
   メイン関数    hen1:5.6   hen2:8.3   hen3:6.8   menseki:?
   関数helon     a:5.6      b:8.3      c:6.8      s:10.35       S:?       test:357.78
   ```

6. 面積 S の計算 (45 行目実行直後).

   ```
   メイン関数    hen1:5.6   hen2:8.3   hen3:6.8   menseki:?
   関数helon     a:5.6      b:8.3      c:6.8      s:10.35       S:18.91   test:357.78
   ```

7. 関数 helon での処理が終了 (48 行目実行) し，制御がメイン関数に戻る (21 行目)．関数 helon で定義した変数は全て自動変数なので，関数 helon での処理が終了するとそれらは消滅する．45 行目の return S; で返される値は，21 行目の関数 helon(hen1，hen2，hen3) の値となり，面積を表す変数 menseki に代入される．

   ```
   メイン関数    hen1:5.6   hen2:8.3   hen3:6.8   menseki:18.91
   ```

8. 面積の表示 (26 行目実行直後).

   ```
   メイン関数    hen1:5.6   hen2:8.3   hen3:6.8   menseki:18.91
   ```

9. プログラムの完了 (29 行目実行直後)．メイン関数の変数は全て消滅する．return 0; でこのプログラムの呼び出し元に戻り値として，0 を返す．このプログラムの呼び出し元は，Linux という OS である．

[1]実行に先立って，プログラムをメインメモリーに格納する動作．この動作については，今は分からなくてもよい．

1.2 三角形の面積と角度の計算

三角形の面積のみならず周長も計算する関数を実装したい場合，先ほどの方法だと困ったことが生じる．複数の戻り値—面積と周長—の値を同時に返せないのである．もちろん，実引数や仮引数を使うこともできない．これらの引数は，呼び出し元から関数にデータを渡すことしかできない．つまり，

- 戻り値で呼び出し元へ返せる値は，一つに限られる．
- 引数は呼び出し元から関数へデータを渡すことはできるが，その逆は不可能．ようするに引数を使ったデータの受け渡しは一方通行である．

である．どうするか?.

いろいろな方法があるが，諸君のこれまでの知識ではグローバル変数を使う方法が良いだろう．リスト 2 に示すように計算結果をグローバル変数に格納する．こうすれば，複数のデータを呼び出し側へ返すことができる．

賢い諸君は，引数もグローバル変数で渡すことができることに気が付くだろう．もちろん，これも可能であるが，よくないプログラム作法である．グローバル変数は，いかなる関数からもアクセス可能であるため，思わぬデータが紛れ込む可能性がある．このようなバグは非常に分かりにくいので，原因を突き止めるのに大変な量力を要する．したがって，グローバル変数はできるだけ使わないようにしなくてはならない．

リスト 2: 三角形の面積と周長を計算するプログラム．

```
1  #include <stdio.h>
2  #include <math.h>
3
4  void info_tri(double a, double b, double c);     // プロトタイプ宣言
5  double S, total_len;                               // グローバル変数
6
7  //============================================================
8  // メイン関数
9  //============================================================
10 int main(void)
11 {
12   double hen1, hen2, hen3;
13   double menseki;
14
15   printf("辺1の長さ?\t");
16   scanf("%lf",&hen1);
17   printf("辺2の長さ?\t");
18   scanf("%lf",&hen2);
19   printf("辺3の長さ?\t");
20   scanf("%lf",&hen3);
21
22   info_tri(hen1, hen2, hen3);
23
24   if(S < -990){
25     printf("入力した辺では，三角形はできません!!!!!\n");
26   }else{
27     printf("面積は，%fです．\n", S);
28     printf("周長は，%fです．\n", total_len);
29   }
30
31   return 0;
32 }
33
```

```
34 //=========================================================
35 // ユーザー定義関数
36 //=========================================================
37 void info_tri(double a, double b, double c)
38 {
39   double s, test;
40
41   s=(a+b+c)/2;
42   test=s*(s-a)*(s-b)*(s-c);
43
44   if(test<=0){
45     S = -999.0;
46   }else{
47     S = sqrt(test);
48     total_len = a+b+c;
49   }
50
51 }
```

1.2.1 グローバル変数の生成と消滅

リスト 2 のローカル変数はすべて自動変数なので，その挙動はリスト 2 と同じである．説明するまでも無いだろう．グローバル変数 S と total_len について，その生成と消滅のタイミングについて述べておく．

- グローバル変数 S と total_len の記憶領域は，プログラムに実行に先立って確保される．
- グローバルへん数はプログラムが終了—メイン関数の return 文の処理が完了した—ときに，消滅する．グローバル変数の記憶領域はプログラムが実行されているときは，常に確保されれている．

2 プログラム作成の練習

[**練習 1**] 図 1 の場合の磁場 (磁界) の強さ H を計算するプログラムを作成せよ．磁界の強さ H はユーザー定義関数で計算する．

- 引数は，電流 I と電流の微小長さ $\Delta\ell$，距離 r，そして角度 θ とする．
- 戻り値は，磁界の強さ H である．

[**練習 2**] 球の半径をあたえると，体積と表面積を計算するプログラムを作成せよ．体積と表面積はユーザー定義関数で計算し，計算結果の 2 つの値はグローバル変数で返す．

[**練習 3**] 先週の練習問題の続き．

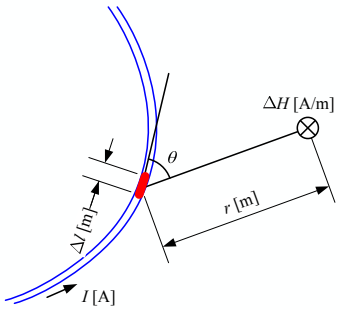

図 1: ビオ・サバールの法則を使う場合の配置.

3　課題

試験明け後の最初の講義の日 (12 月 15 日) の AM8:45 までに，以下の課題をレポートとして提出すること．表紙等は，いつもの通り．表紙のタイトルは「関数 (追加説明)」とすること．

[**問 1**]　(復) 後期中間試験で分からなかったところの問題を解いて，きちんとした解答を作成せよ．

[**問 2**]　(予) 教科書 p.204–223 を 3 回，繰り返し読め．

[**問 3**]　(予) 配列と呼ばれるデータ構造について，以下をまとめよ．

- 配列を使う理由
- 配列の定義の仕方