# 制御文(switchとif〜else if〜else)

山本昌志*

2006年7月19日

**概 要**

条件により，複数の実行文を選択する多分岐について，学習する．switchとif〜else if〜elseの文法を理科して，練習問題でその使い方をマスターする．

## 1 本日の授業内容

### 1.1 前回の復習

先週は教科書 [1] の 4 章 制御の流れの p.102–117 を学習した．そこでは，関係演算子や論理演算子，if〜else 文の学習を行った．先週の学習内容は以下の通りで，絶対に理解し，憶えなくてはならない．

- 正しいあるいは誤りを整数の 0 と 1 で表す．正しい場合が整数の 1 で，誤りの場合が整数の 0 である．コンピューターは，なんでもかんでも数字で表すのである．C 言語では特別に，0 を誤り，それ以外—大体の場合 1— を正しいとして取り扱う．
- 関係演算子は大小関係や等しいか否かについて，計算を行う．これは四則演算子 (+,-,*,/) と同じ演算子なので，計算結果がある．それは，正しい場合 (真) に 1，誤り (偽) の場合 0 となる．
  - 大小関係を表す演算子は 4 つ (<, <=, >, >=) ある．それぞれ，数学の記号の $<$，$\leqq$，$>$，$\geqq$ に対応する．キーボードに $\leqq$ の記号がないので，<=で代用しているのである．ただし，=<と書いてはならない．
    - 例えば，演算 3>5 の結果は 0 となる．なぜならば，この式が示している大小関係は誤りだからである．
    - 一方，演算 3<5 の結果は 1 となる．なぜならば，この式が示している大小関係は正しいからである．
  - 大小関係を示す関係演算子とともに，等しいか否かを表す演算子も重要な役割を果たす．それには，等しいことを表す==と等しくないことを表す!=がある．これらは，それぞれ，数学記号の $=$ と $\neq$ に対応する．イコールひとつだと代入演算子になるので，イコールをふたつ続けて等しいことを表す演算子として使用している．$\neq$ はキーボードに記号がないので，!=を使っている．

*独立行政法人秋田工業高等専門学校電気工学科

    - 例えば，演算 3==5 の結果は 0 となる．なぜならば，この式が示している等価関係は誤りであるからである．
    - 一方，演算 3!=5 の結果は 1 となる．なぜならば，この式が示している等価関係は正しいからである．

- 論理演算子は論理が正しいか否かについて，演算を行う．演算結果は，論理が正しければ 1，誤りであれば 0 となる．論理演算子には，論理和 || と論理積 &&，論理否定 ! の 3 つがある．
  - 論理和は，日本語では「または」英語では「or」と表現される．論理和演算子をはさむ 2 つの論理のどちらか一方が正しい，あるいは両方が正しい場合，演算結果が 1 となる．両方の演算が誤りの場合のみ 0 となる．
    - 例えば，演算 9<7 || 5<3 の結果は 0 となる．なぜならば，演算子||の両側の式は誤りであるからである．
    - 次に，演算 9<7 || 3<5 の結果は 1 となる．なぜならば，論理和演算子をはさむ片方の式が正しいからである．
    - 言うまでもないが，演算 7<9 || 3<5 のように，両方の式が正しい場合も，演算の結果は 1 となる．
  - 論理積は，日本語では「かつ」英語では「AND」と表現される．論理積演算子をはさむ 2 つの論理の両方の演算が正しい場合のみ 1 となる．どちらか一方が誤り，あるいは両方が誤りの場合，演算結果が 0 となる．
    - 例えば，演算 3<5 && 7<9 の結果は 1 となる．なぜならば，演算子||の両側の 2 つの式は正しいからである．
    - 次に，演算 3<5 || 9<7 の結果は 0 となる．なぜならば，論理和演算子をはさむ片方の式が誤りだからである．
    - 言うまでもないが，演算 5<3 || 9<7 のように，両方の式が誤り場合も，演算の結果は 0 となる．
  - 論理否定は，論理を反転させる．英語では,「NOT」と表現される．
    - 例えば，演算 !(3<5) の結果は 0 となる．なぜならば， 3<5 の演算の結果は 1，そして否定演算子 ! でそれを反転させているので，!(3<5) の演算結果は 0 となる．
    - 一方，演算 !(5<3) の結果は 1 となる．なぜならば， 5<3 の演算の結果は 0，そして否定演算子 ! でそれを反転させているので，!(5<3) の演算結果は 1 となる．

- これまで，学習してきた演算子には，算術演算子 (+,-,*,/,%) と関係演算子 (<, <=, >, >=)，論理演算子 (||,&&,!) がある．これらの演算子 (オペレーター) と被演算子 (オペランド )[1]を組み合わせて式ができあがる．数学同様，演算子には優先順位があり，先に計算する演算子が決まっている．表 1 の上の演算子から計算を行う．

[1]普通，値や変数がオペランドになる．

表 1: 演算子の優先順位 (上のほうが優先順位が高い)

| 種類 | 演算子 |
|---|---|
| 括弧 | () |
| 論理否定 | ! |
| 乗除 | * / |
| 加減 | + - |
| 比較 | < <= > >= |
| 等価 | == != |
| 論理積 | && |
| 論理和 | \|\| |

- 「もし○○ならば，△△する」という構文—とくに△△の部分が 1 つの文—は次のように書く．

```
if(a<=10) printf("a は，10 以下です\n");
```

- 「もし○○ならば，△△し，□□し，···」のように複数の文を実行する場合は，次のように書く．

```
if(0<=a && a<=10){
  printf("a は，0 以上\n");
  printf("かつ\n");
  printf("a は，10 以下です\n");
}
```

実行させたい複数の文は，括弧 { } でくくり，ブロック化するのがこつである．教科書ではこれを複文と言っている．

- 「もし○○ならば△△し，さもなければ□□する」というように，条件により二者択一の選択処理は，次のように書く．

```
if(0<=a && a<=10){
  printf("a は，0 以上\n");
  printf("かつ\n");
  printf("a は，10 以下です\n");
}else{
  printf("a は，0 未満\n");
  printf("または\n");
  printf("a は，10 より大きい\n");
}
```

## 1.2 本日の学習内容

先週は，条件式 (制御式) により，プログラムが 2 つに分岐する構文を学習した．本日は，さらに進んで，条件により，3 つ以上に分岐する構文を学習する．教科書 [1] の 4 章の p.117–131 の学習を行う．そこで，

諸君が身に付けるべきことを以下に示す.

- switch を使った多分岐のプログラムが書ける．条件式の値が整数の場合，switch 文を使う.
- if〜else if〜else を使った多分岐のプログラムが書ける．条件式の値が整数でない場合，この構文をつかう.

# 2 switch文

プログラム中で,「もし a=1 ならば○○する，a=2 ならば△△する，a=5 ならば□□する，さもなければ◎◎する」というような処理をしたい場合，switch という命令を使う．このように，値により処理が複数に分岐することを多分岐と言う．条件の部分が整数 (int)，あるいは文字 (char) で表される場合，switch を使う．

## 2.1 プログラム例

リスト 1 に switch 文を使った例を示す．その動作は，以下のとおりである．

1. 「あなたは何年生ですか?」と質問を行い，ユーザーはそれに答える．答えた結果は，変数 year に格納する．
2. 変数 year の値に従い，以下の動作をする．
   - **year が 1 の場合**「高専の生活にも慣れ，少し余裕ができる.」と表示する．
   - **year が 2 の場合**「留年しなかったので，油断する.」と表示する．
   - **year が 3 の場合**「レポートがきついが，要領を憶える.」と表示する．
   - **year が 4 の場合**「専門科目が多く，たいへんきつい.」と表示する．
   - **year が 5 の場合**「進路も決まり，ほっと一息.」と表示する．
   - **year が 1–5 以外の場合**「?年生はありません!」と表示する．

このような構文は，先週，学習した if 文を使っても可能であるが，わかり難いプログラムになる．switch 文を使う方が簡単である．

リスト 1: switch 文を使った秋田高専の学生の心境表示プログラム

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5    int year;
6
7    printf("あなたは何年生ですか?\t");
8    scanf("%d",&year);
9
10   switch(year)
11     {
12     case 1:
13       printf("高専の生活にも慣れ，少し余裕ができる．\n");
14       break;
15     case 2:
16       printf("留年しなかったので，油断する．\n");
17       break;
18     case 3:
19       printf("レポートがきついが，要領を憶える．\n");
20       break;
21     case 4:
```

```
22          printf("専門科目が多く，たいへんきつい．\n");
23          break;
24        case 5:
25          printf("進路も決まり，ほっと一息．\n");
26          break;
27        default:
28          printf("%d年生はありません!\n",year);
29          break;
30        }
31
32      return 0;
33  }
```

## 2.2 switch の書き方

if 文は，選択肢が少ない場合，わかりやすい記述ができる．しかし，選択肢が多くなると，記述は複雑になり，分かりにくいプログラムとなる．そのような場合は，if 文の代わりに switch 文を使う．switch 文を使った多分岐の書式は，次のとおりである．マッチした定数式の部分から実行される．

書式

```
switch(式){
  case 定数式1:
    文1;
    文2;
    break;
  case 定数式2:
    文1;
    文2;
    break;
  case 定数式3:
    文1;
    文2;
    break;
  default:
    文7;
    文8;
    break;
}
```

式や定数式の値の型は，int または char でなくてはならない[2]．とくに，定数式は，コンパイル時に，値が評価できなくてはならない．case の後の定数式は，ラベルと呼ばれるものである．ラベルの後は，コロン (:) をつける．文の終わりを示すセミコロン (;) ではない．実行する文の集まりの最後には，bread 文を書く．この break 文が無いと，マッチしたラベル以降の全ての文が実行される．コードブロックを表す中括弧 { } が無いので，break 文で switch 文の終わりを示す最後の中括弧 } から抜け出す．

[2]列挙型も使えるが，これは後の学習範囲である．

この構文のフローチャートを，図 1 に示す．これは，式の値により，それにマッチしたブロック[3]が実行される．もし，どれもマッチしなければ，default が実行される．default 文は無くてもよいが，その場合はどのブロックも実行されない場合がある．

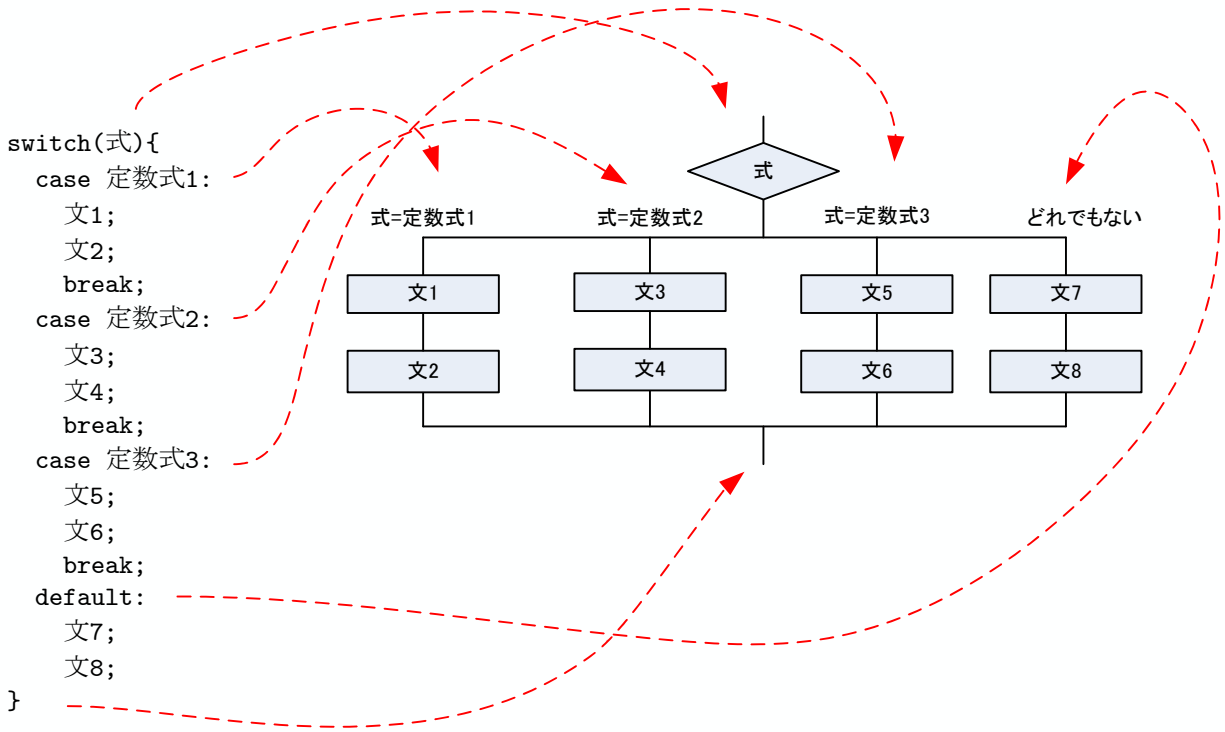


図 1: switch 文を使った構文．多くの選択肢がある場合．

## 3 if〜else if〜else 文

「もし○○ならば ⊛⊛ する．さもなければ，もし□□ならば ⊠⊠ する．さもなければ，もし △△ ならば ▽▽ する．さもなければ，◎◎ する.」というよう構文を書きたい場合がある．条件に合致しなければ，次々と条件を変化させる．このような構文には，if〜else if〜else 文を使う．

### 3.1 プログラム例

リスト 2 に，if〜else if〜else を使った例を示す．動作は説明しなくても分かるだろう．温度を入力すると，それに従い「極寒です」,「寒いです」,「快適です」,「暑いです」,「猛暑です」と表示する．これは，

[3]コードブロックではないので，中括弧 ({}) がない．

先ほどの switch 文では不可能である．なぜならば，温度の変数 temp は倍精度実数で，かつ値の範囲により動作内容が異なるからである．

リスト 2: if〜else if〜else を使った気温の感じ方表示プログラム

```
1  #include <stdio.h>
2
3  int main(void)
4  {
5    double temp;
6
7    printf("今の気温は?\t");
8    scanf("%lf",&temp);
9
10   if(temp<-30){
11     printf("極寒です\n");
12   }else if(-30 <= temp && temp < 10){
13     printf("寒いです\n");
14   }else if(10 <= temp && temp < 25){
15     printf("快適です\n");
16   }else if(25 <= temp && temp < 35){
17     printf("暑いです\n");
18   }else{
19     printf("猛暑です\n");
20   }
21
22   return 0;
23 }
```

## 3.2 文の書き方

if と else if，else を使った多分岐の書式は，次のとおりである．最初にマッチしたブロックのみ実行される．

書式

```
if(制御式 1){
    文 1;
    文 2;
}else if(制御式 2){
    文 3;
    文 4;
}else if(制御式 3){
    文 5;
    文 6;
}else{
    文 7;
    文 8;
}
```

これは,「制御式 1 が正しい (真) ならば，文 1 と文 2 を実行する．さもなければ，制御式 2 が正しいならば，文 3 と文 4 を実行する．さもなければ，制御式 3 が正しいならば，文 5 と文 6 を実行する．さもなければ，文 7 と文 8 を実行する.」となる．

この構文のフローチャートを，図 2 に示す．このフローチャートを見てわかるように，最初に真となった制御式に続くブロック内が実行されのみである．それ以降，真になっても，そのブロックは実行されない．どの制御式も真にならない場合，最後の `else` のブロックが実行される．即ち，実行されるブロックは 1 個のみである．`else if` の段数をいくらでも増やせることは，言うまでもない．

また，`else` が無い構文も許される．この場合，真となる制御式がない場合，どのブロックも実行されず，この構文から抜ける．

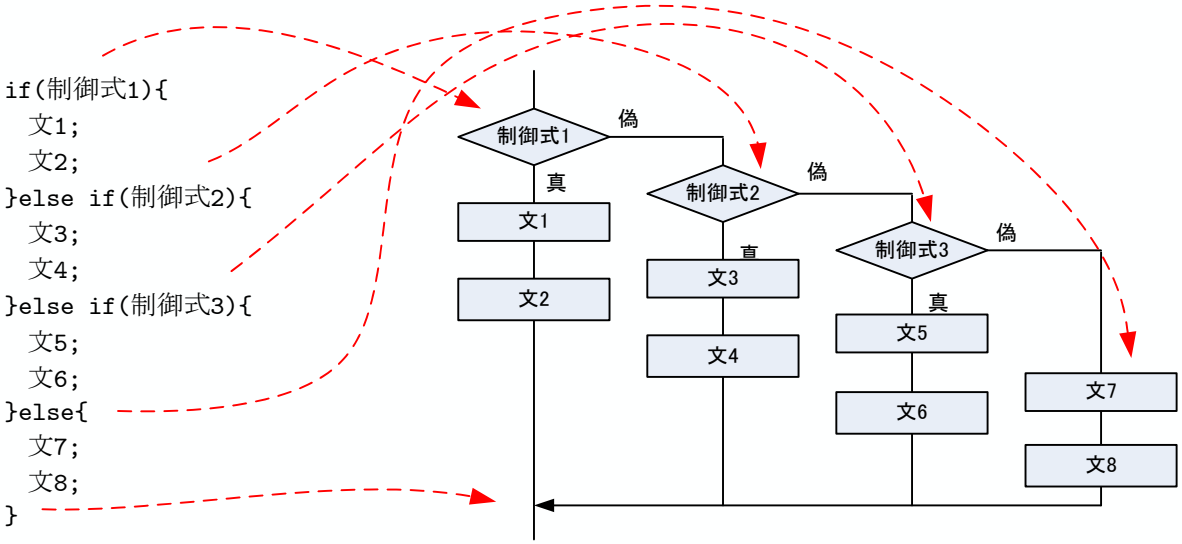


図 2: `if`〜`else if`〜`else` を使った多分岐

## 4　プログラム作成の練習

本日の学習内容の理解を深めるために，以下の練習問題のプログラムを作成して，実行してみよ．最初の 2 問は `switch` 文を使うこと．最後の問は，`if`〜`else if`〜`else` を使うこと．

**[練習 1]** クイズを出題して，結果を判定するプログラムである．以下のようなプログラムを作成せよ．

- 画面に「情報処理基礎の担当教員は?」と表示する．
- 画面に「1:山田　2:山上　3:山本」と表示する．
- キーボードから，整数値を読み込み，変数に格納する．

– 読み込んだ整数に従い，以下を実行する．
  * 1 の場合
    - 「山田ではありません.」と表示する．
    - 「不正解です.」と表示する．
  * 2 の場合
    - 「山上ではありません.」と表示する．
    - 「不正解です.」と表示する．
  * 3 の場合
    - 「情報処理基礎の担当は，山本です.」と表示する．
    - 「正解です.」と表示する．
  * 1–3 以外の場合
    - 「質問にまじめに答えろ.」と表示する．

**[練習 2]** 図 3 に示す 4 種類の回路がある．全ての抵抗は同一とする．キーボードから，ひとつの抵抗の抵抗値 [Ω] と電圧 [V] を入力する．そして，計算する回路を選択する．回路の全抵抗 [Ω] と発熱量 [W] を出力するプログラムを作成せよ．

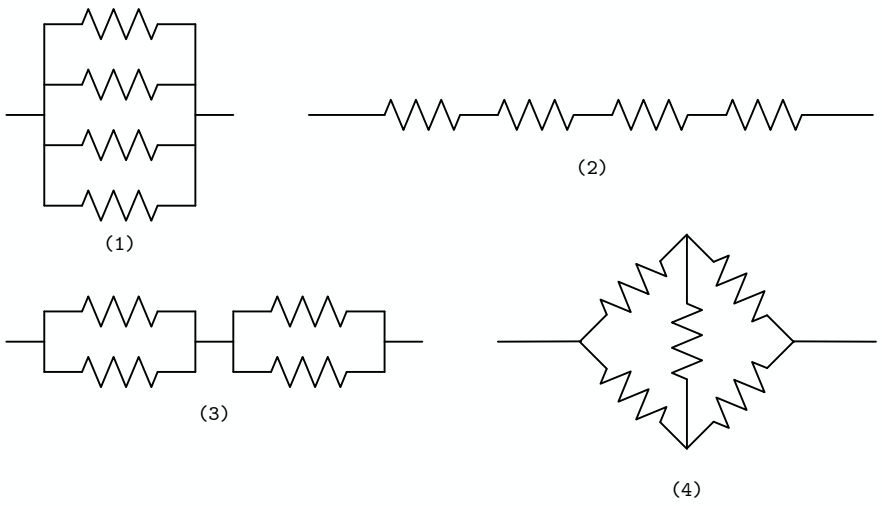


図 3: 同じ抵抗から構成される 4 つの回路.

**[練習 3]** 三角形の面積を計算するプログラムを作成せよ．入力データは，三辺の長さ $(a, b, c)$ とする．三辺の長さから，三角形の面積はヘロンの公式を使って，

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

と計算できる．$S$が三角形の面積になる．このプログラムを使うユーザーはおっちょこちょいなので，入力データのチェックを行う以下のプログラム—ルーチン—を付加する．

- もし，辺の長さ`a`, `b`, `c`のいずれかが負の場合,「辺の長さが負です」と表示する．そして，プログラムは終了—`return`文を実行—する．
- もし，いずれかの辺の長さが，他の 2 辺の和よりも大きいならば,「三角形はできません」と表示する．そして，プログラムは終了する．
- 上の 2 つの条件に当てはまらなければ，ちゃんとした三角形ができる．この場合は，面積を計算して，ディスプレイに出力する．

## 参考文献

[1] 内田智史監修, (株) システム計画研究所編. C 言語によるプログラミング　基礎編　第 2 版. (株) オーム社, 2006.