

if 文...条件分岐処理

三池 克明

if 文は条件の判定から処理を分岐させることができます。
これにより、状況に応じて処理内容の変更ができるようになります。

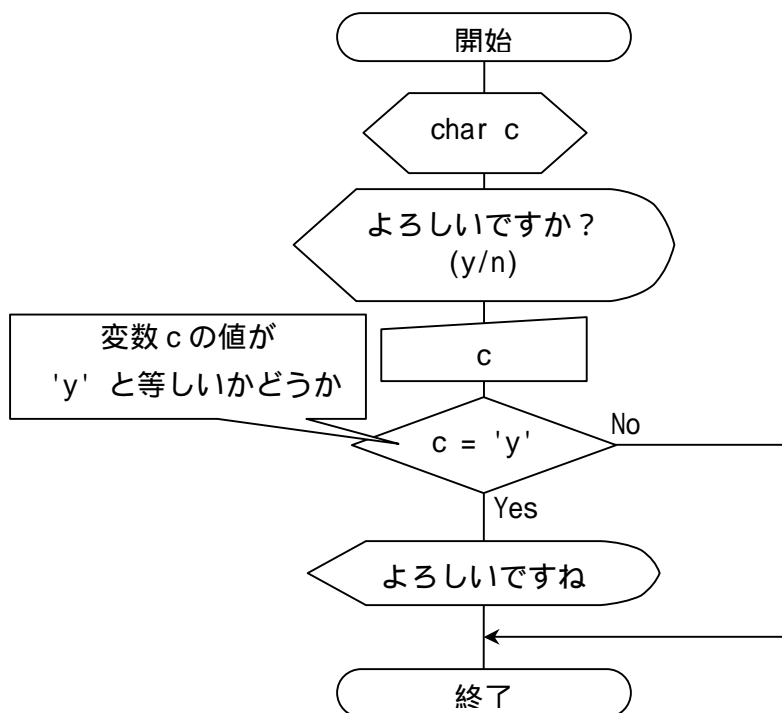
目 次

1.	if ~	1
1.1.	確認を促す	1
1.2.	条件式で扱える演算子	4
2.	if ~ else ~	5
2.1.	確認を促す、その2	5
2.2.	合否の判定	8
3.	if ~ else if ~	10
3.1.	確認を促す、その3	10
3.2.	合否の判定、その2	13
4.	演習問題	15

1. if~

1.1. 確認を促す

まずは簡単な分岐プログラムを作ってみましょう。
以下はそのフローチャートです。



if1.cpp

```
1: #include <stdio.h>
2:
3: main()
4: {
5:     char    c;
6:
7:     printf("よろしいですか? (y/n) : ");
8:     scanf("%c", &c);
9:
```

```
10:         if(c == 'y'){
11:             printf("よろしいですね。¥n");
12:         }
13:     }
```

実行例 1

```
よろしいですか? (y/n) : y
よろしいですね。
```

実行例 2

```
よろしいですか? (y/n) : n
```

実行例 3

```
よろしいですか? (y/n) : Y
```

このように「y」を入力した場合は「よろしいですね。」と表示され、そうでなければ何もせずに終了します。

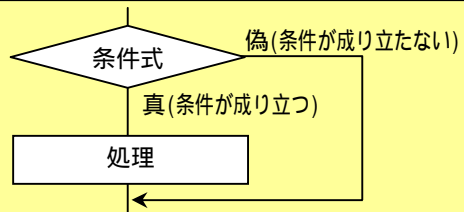
この処理を行っているのは以下の部分です。

```
10:         if(c == 'y'){
11:             printf("よろしいですね。¥n");
12:         }
```

また、今回はじめて使用した if 文の構文は以下のとおりです。

構文：

```
if(条件式){
    処理
}
```



このように条件式が真（成り立つ）ときだけ中カッコ{ }内の処理を行わせることができます。

ここでいう条件式とは数学的・論理的なことから表現する式で、if2.cppの10行目にある「c == 'y'」は「変数cの値は'y'と等しい」という意味になります。

数学的には「c = 'y'」ですが、=は代入演算子で使用しているためC言語では「==」を使います。

続いて以下のソースプログラムを入力し、コンパイル、実行してみましょう。

if2.cpp

```
1: #include <stdio.h>
2:
3: main()
4: {
5:     char    c;
6:
7:     printf("よろしいですか?(y/n) : ");
8:     scanf("%c", &c);
9:
10:    if(c == 'y' || c == 'Y'){
11:        printf("よろしいですね。¥n");
12:    }
13: }
```

実行例 1

```
よろしいですか?(y/n) : y
よろしいですね。
```

実行例 2

```
よろしいですか?(y/n) : Y
よろしいですね。
```

今度は「y」か「Y」が入力されたときだけ「よろしいですね。」と表示されるようになりました。

1.2. 条件式で扱える演算子

条件式で扱える演算子を表にまとめました。
必要に応じて参照するようにしましょう。

分類	演算子	意味	読み方	使用例
関係演算子	==	= (等しい)	イコール イコールイコール	A == B (AとBは等しい)
	!=	(等しくない)	ノットイコール	A != 0 (Aは0と等しくない)
	<	< (より小さい)	小なり	A < B (AはBより小さい)
	>	> (より大きい)	大なり	A > 0 (Aは0より大きい Aは正の数)
	<=	(以下)	小なりイコール	A <= 0 (Aは0以下)
	>=	(以上)	大なりイコール	A >= 80 (Aは80以上)
論理演算子	&&	かつ	アンド アンドアンド	A == 1 && B == 0 (Aが1であり且つBが0)
		または	オア オアオア	A == 1 B == 0 (Aが1である、またはBが0)
	!	否定	ノット ビックリ イスクラメーション	!(A == 0) (A=0ではない A 0)

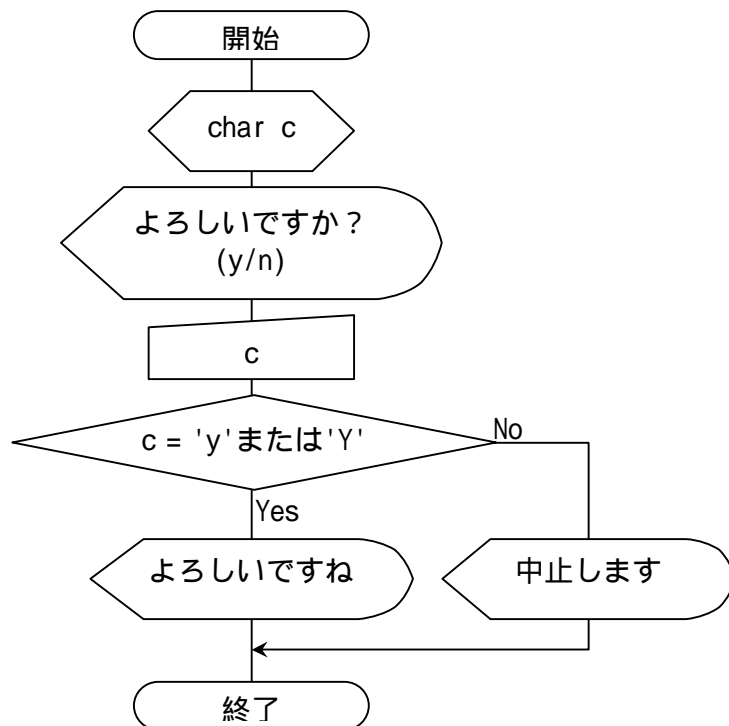
if2.cppの10行目の条件式「c == 'y' || c == 'Y」は「変数cは'y'と等しい、または変数cは'Y'と等しい」となりますね。

もっとわかりやすく表現すれば、「変数cの値は'y'あるいは'Y'である」という意味になります。

2. if ~ else ~

2.1. 確認を促す、その2

if2.cpp を少しだけ改良しました。
以下がそのフローチャートです。



そしてそのソースプログラムです。
入力し、コンパイル、実行してみましょう。

if3.cpp

```
1: #include <stdio.h>
2:
3: main()
4: {
5:     char    c;
```

```
6:
7:     printf("よろしいですか?(y/n) : ");
8:     scanf("%c", &c);
9:
10:    if(c == 'y' || c == 'Y'){
11:        printf("よろしいですね。¥n");
12:    } else {
13:        printf("中止します。¥n");
14:    }
15: }
```

実行例 1

```
よろしいですか?(y/n) : Y
よろしいですね。
```

実行例 2

```
よろしいですか?(y/n) : N
中止します。
```

実行例 3

```
よろしいですか?(y/n) : a
中止します。
```

このように「Y(または y)」を入力すれば「よろしいですね。」と表示され、そうでなければ「中止します。」と表示されるようになりました。

ですから実行例 3 のように Y でも N でもない文字を入力しても「中止します。」と表示されます。

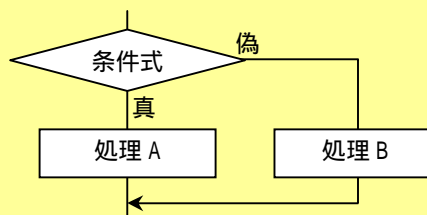
これを問題と捉えるか捉えないかは状況によりまちまちですので本書では判断できません。

とりあえず「コンピュータは逐一指示してやらないといけない」と理解しておきましょう。

なお、このプログラムで使った if 文の構文は以下のとおりです。

構文：

```
if(条件式){  
    処理 A  
} else {  
    処理 B  
}
```

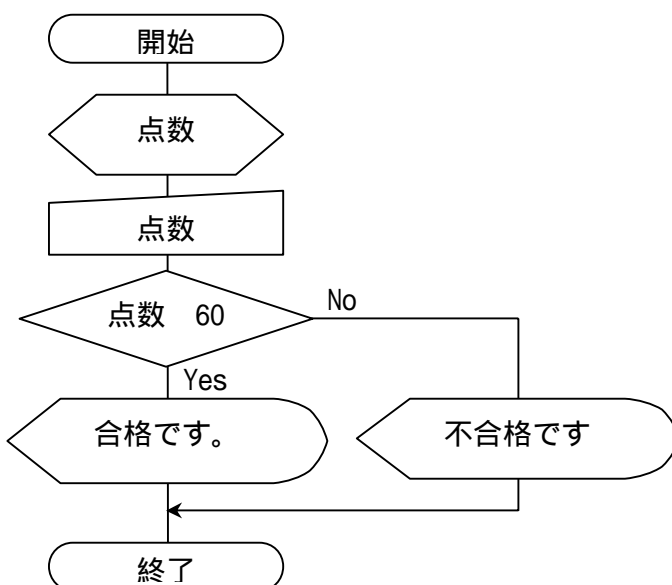


2.2. 合否の判定

今度は点数から合否を判定するプログラムです。

以下はその変数リストとフローチャート、そしてソースプログラムです。

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
点数	score	int



if4.cpp

```
1: #include <stdio.h>
2:
3: main()
4: {
5:     int          score;
6:
7:     printf("点数を入力してください : ");
8:     scanf("%d", &score);
9:
10:    if(score >= 60){
11:        printf("合格です。¥n");
12:    } else {
```

```
13:             printf("不合格です。¥n");
14:         }
15:     }
```

実行例 1

点数を入力してください : 70
合格です。

実行例 2

点数を入力してください : 50
不合格です。

実行例 3

点数を入力してください : 12245
合格です。

実行例 4

点数を入力してください : -134
不合格です。

このように点数が 60 以上かそうでないかで「合格です」あるいは「不合格です」が表示されるようになりました。

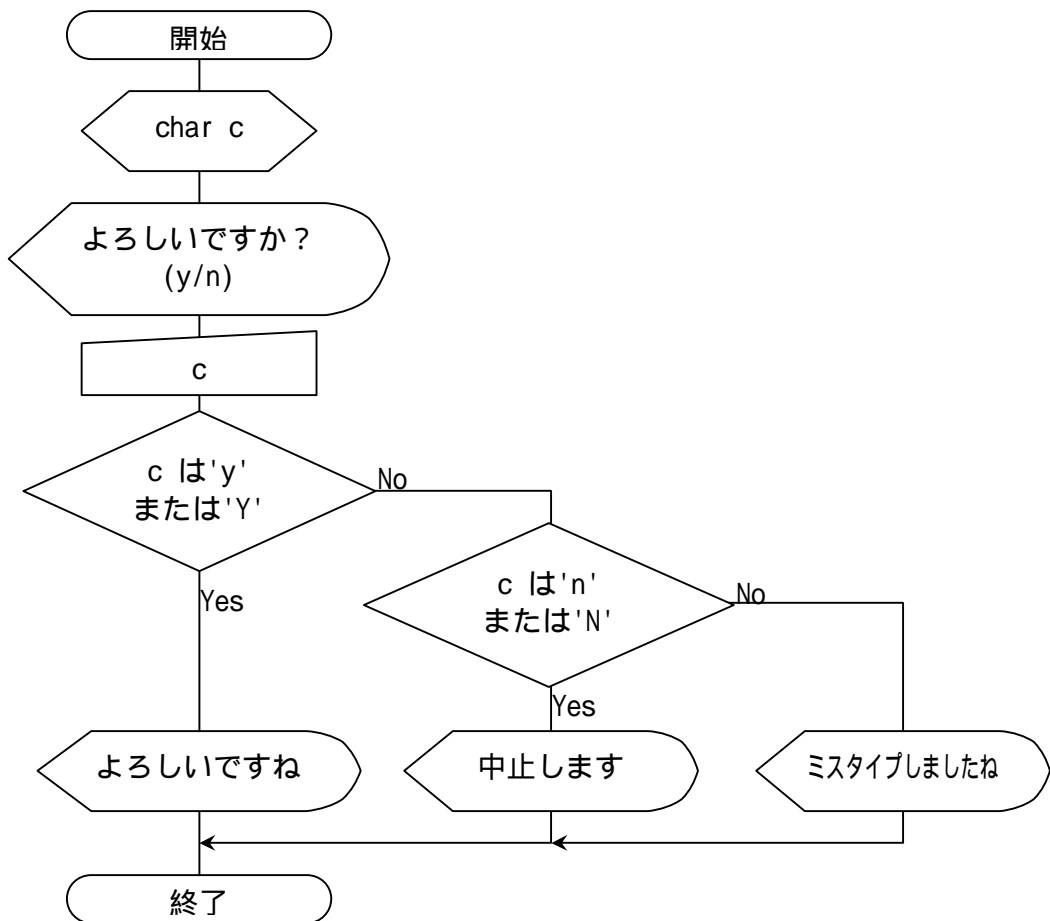
ですから点数が 100 を越えていようと、0 を下回ろうとお構いなしに判定をします。

3. if ~else if ~...

3.1. 確認を促す、その3

if3.cpp を改良しました。

以下はそのフローチャートとソースプログラムです。



```
1: #include <stdio.h>
2:
3: main()
4: {
5:     char    c;
6:
7:     printf("よろしいですか？(y/n) : ");
8:     scanf("%c", &c);
9:
10:    if(c == 'y' || c == 'Y'){
11:        printf("よろしいですね。¥n");
12:    } else if(c == 'n' || c == 'N'){
13:        printf("中止します。¥n");
14:    } else {
15:        printf("ミスタイプしましたね。¥n");
16:    }
17: }
```

実行例 1

```
よろしいですか？(y/n) : y
よろしいですね。
```

実行例 2

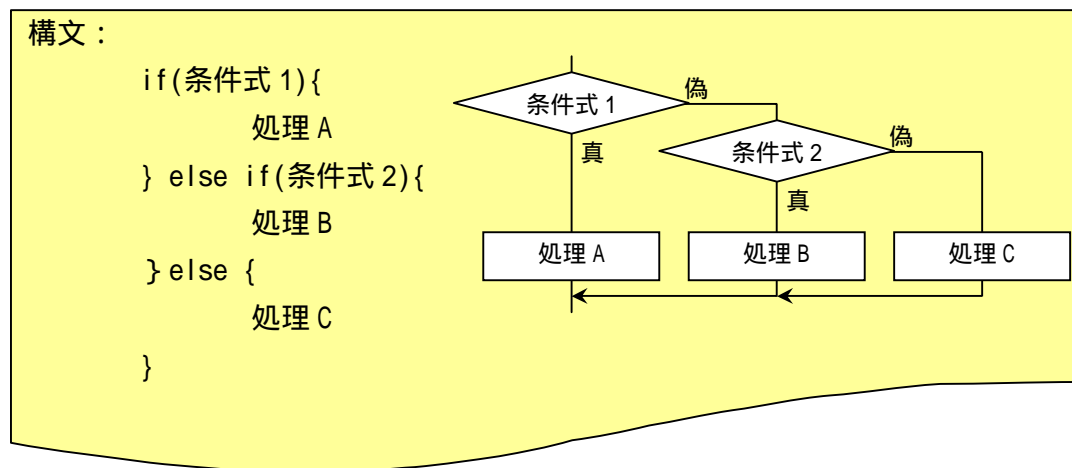
```
よろしいですか？(y/n) : N
中止します。
```

実行例 3

```
よろしいですか？(y/n) : x
ミスタイプしましたね。
```

今回はY、N、その他の文字で処理を分岐させました。
このように if 文は2つだけでなく3つ以上の分岐処理も可能です。

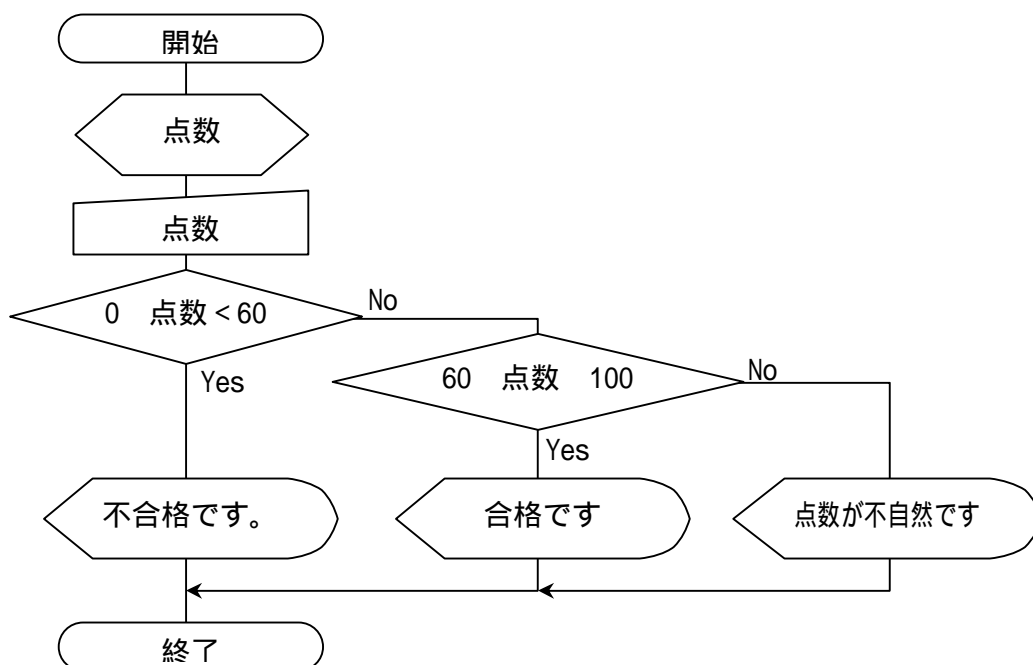
なお、このプログラムで使った if 文の構文は以下のとおりです。



3.2. 合否の判定、その2

if4.cpp を改良しました。

以下はそのフローチャートとソースプログラムです。



if6.cpp

```
1: #include <stdio.h>
2:
3: main()
4: {
5:     int         score;
6:
7:     printf("点数を入力してください : ");
8:     scanf("%d", &score);
9:
10:    if(0 <= score && score < 60){
11:        printf("不合格です。¥n");
12:    } else if(60 <= score && score <= 100){
13:        printf("合格です。¥n");
```

```
14:         } else {
15:             printf("点数が不自然です。¥n");
16:         }
17:     }
```

実行例 1

点数を入力してください : 70
合格です。

実行例 2

点数を入力してください : 50
不合格です。

実行例 3

点数を入力してください : 125
点数が不自然です。

今回は 0 ~ 59 なら不合格、60 ~ 100 なら合格、そしてそれ以外なら不自然と判定するようにしました。

これにより 0 より小さい、あるいは 100 より大きい点数にも対応できるようになりました。

4. 演習問題

4-1. 試験の点数を入力させてから A~D の 4 段階評価を判定するプログラム「ensyu4 -1.cpp」を作りなさい。また以下の仕様に従うこと。

- 0~59 点は D 判定、60~69 点は C 判定、70~79 点は B 判定、80~100 点は A 判定とする。
- 点数は int 型で入力させる
- 点数が不自然な場合は警告メッセージを表示する

実行例 1

点数を入力してください : 79
判定は B です。

実行例 2

点数を入力してください : -20
点数が不自然です。

4-2. 二次方程式 $Ax^2 + Bx + C = 0$ がある。各定数 A~C を入力させ二方程式を表示し、さらに判別式から解の種類を表示するプログラム「ensyu4 -2.cpp」を作りなさい。なお A、B、C は全て整数とする。

実行例 1

x の二次方程式 $Ax^2 + Bx + C = 0$ の A、B、C を入力してください。
ただし A、B、C は全て整数とします。

A = 2

B = 16

C = 1

方程式は $2x^2 + 16x + 1 = 0$ ですね。

異なる 2 つの実数解です。

実行例 2

x の二次方程式 $Ax^2+Bx+C=0$ の A、B、C を入力してください。

ただし A、B、C は全て整数とします。

$$A = 4$$

$$B = 4$$

$$C = 1$$

方程式は $4x^2+4x+1=0$ ですね。

重解です。

実行例 3

x の二次方程式 $Ax^2+Bx+C=0$ の A、B、C を入力してください。

ただし A、B、C は全て整数とします。

$$A = 2$$

$$B = 3$$

$$C = 4$$

方程式は $2x^2+3x+4=0$ ですね。

異なる 2 つの虚数解です。

ヒント：

二次方程式 $ax^2 + bx + c = 0$ の判別式は

$$D = b^2 - 4ac$$

でしたよね。

また、

$D < 0$ なら「異なる 2 つの虚数解」

$D = 0$ なら「重解」

$D > 0$ なら「異なる 2 つの実数解」

でしたよね。