第5回条件による分岐

1

今回の目標

- 式、文（単文、ブロック）を理解する。
- 条件分岐の仕組みを理解する。
- 関係演算子、論理演算子の効果を理解する。

☆2次方程式の解を求めるプログラムを作成する。

2

2次方程式の解法

2次方程式 $ax^2 + bx + c = 0$ は、

判別式(discriminant) $D = b^2 - 4ac \geq 0$ のとき、

実数解

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

を持つ。

判別式 $D = b^2 - 4ac < 0$ のとき、

実数解を持たない。

3

式と単文

C言語では、

式：定数、変数、関数呼び出し
と、それらを演算子で結合したもの。

式の例

```
3.14
age=20
radius * radius
area =  3.14 * radius * radius
```

単文：式＋';'

単文の例

```
3.14;
age=20;
radius * radius;
area =  3.14 * radius * radius;
```

4

文と複文

文：単文、複文、・・・

単文

```
******;
```

複文
（ブロック）

```
{
    ****;
    *********;
}
```

文をならべて、
中括弧で囲んだもの。

C言語のプログラムは、
このような文（単文、複文、・・・）から構成される。

5

複文とインデント

```
{
    ****;
    *********;
}
```

中括弧は、
それだけを書く事。

中の文は、
一段字下げして
左端をそろえる事。

中括弧とじは、
対応する中括弧と
同じ列に書く事。

（スタイル規則参照）

6

if文

C言語で、条件式によって、
文を選択して実行する文

書式

```
if(条件式)
    {
        選択実行部分1
    }
```

条件式が **真** なら選択実行部分1を **実行する**。

条件式が **偽** なら選択実行部分1を **実行しない**。

7

式と真偽

C言語には真と偽を表す専用の型はなく、
int型の値で代用する。

真：1（0以外）

偽：0

if文の条件式には、
真偽を表す整数型の式（論理式）
を書く。
（スタイル規則参照）

必ず、中括弧を書く。
（スタイル規則参照）

この例では、
この部分は実行
されます。

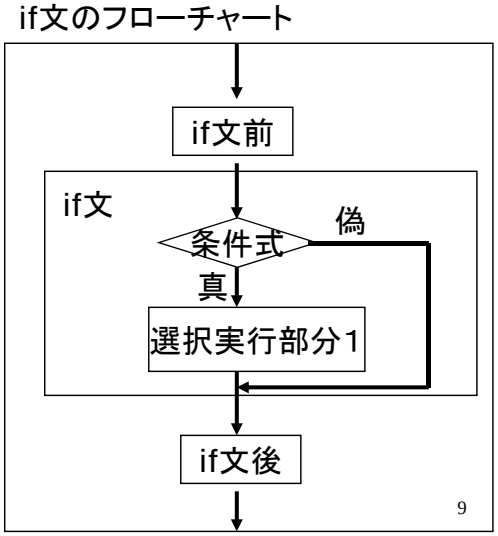
```
int   bool;

bool=1;
if(bool)
   {
      …
   }
```

8

if文の動作1（フローチャート）

書式

```
if(条件式)
    {
        選択実行部分1
    }
```



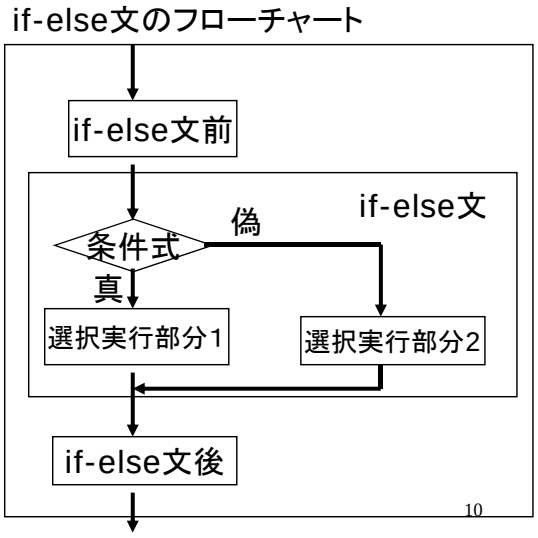
9

if-else文

条件によって2つの文のどちらかを選択して実行する。

書式

```
if(条件式)
    {
        選択実行部分1
    }
else
    {
        選択実行部分2
    }
```



10

練習1

```
/*if_test.c      コメント省略  */
#include<stdio.h>
int  main()
{
       int  a;
       printf("実験開始 ¥n");
       if(1)
          {
             printf("常に表示される。¥n");
          }

       if(0)
          {
             printf("常に表示されない。¥n");
          }
/*次のページに続く*/
```

11

```
/*前ページの続き */
       printf("1(真)または0(偽)を入力して下さい。¥n");
       scanf("%d", &a);

       if(a)
          {
             printf("真です。aの値は0以外です。¥n");
          }
       else
          {
             printf("偽です。aの値は0です。¥n");
          }

       printf("実験終了¥n");
       return  0;
}
```

12

関係演算子

a = = b	a がbと等しい時に真
a ! = b	a がbと等しくない時に真
a < b	a がbより真に小さいとき真
a > b	a がbより真に大きいとき真
a < = b	a がb以下のとき真
a > = b	a がb以上のとき真

関係演算子を使った式は、真偽値を表すint型の値を返す。
本演習では、関係演算子を使った式は論理式として扱い、
算術式とは明確に区別すること。

13

間違いやすい関係演算子（1）

間違い　a =< b　　NG
　　a => b

正しい　a < = b　a がb以下のとき真　　OK
　　a > = b　a がb以上のとき真

他の間違い
a < b ＜ c　NG
a = b ＞ c

関係演算子は2項演算子です。
関係演算子は組み合わせて
使ってはいけません。

これらは、コンパイルエラー
にならない。

14

間違いやすい関係演算子（2）

関係演算子「＝＝」と代入演算子「＝」は間違えやすいので、気をつける事。

「=」は代入演算子。
間違いだが
コンパイルエラーにならない

```
if(a = b)
   {
      printf("同じ数字です。¥n");
   }
```

NG

こう書くと、bの値が0以外のときに実行されます。

if文の条件式には論理式（関係演算子を使った式）を書くこと。

```
if(a==b)
   {
      printf("同じ数字です。¥n");
   }
```

OK

15

関係演算子と型

関係演算子は2項演算子です。
両辺の型を一致させること。
（スタイル規則参照）

比較の左辺は実数なのに、
右辺は整数

```
double  a;
if(a <= 0)
   {
      …
```

NG

```
double  a;
if(a <= 0.0)
   {
      …
```

OK

16

練習2

```
/*relation.c      関係演算子実験( コメント省略 ) */
#include<stdio.h>
int  main()
{
        int a;
        int b;
        printf("2つの整数を入力して下さい¥n");
        scanf("%d", &a);
        scanf("%d", &b);
        if(a==b)
           {
               printf("同じ数字です。¥n");
           }
        else
           {
               printf("異なる数字です。¥n");
           }
        return  0;
}
```

17

論理演算子(1)

演算子	演算の意味	演算結果
!A	A の否定 (NOT A)	A が真のとき !A は偽。 A が偽のとき !A は真。
A && B	AかつB (A AND B)	A と B が共に真のとき A&&B は真。 それ以外のときは偽。
A || B	AまたはB (A OR B)	A と B が共に偽のとき A||B は偽。 それ以外のときは真。

論理演算子の被演算項(AやB)
は論理式だけを記述する。
よって、AやBは真偽値を表す。

18

論理演算子（2）

式1 ＆＆ 式2 ＆＆ ・・・＆＆式n

式1から式nまですべてが真なら真。
それ以外の場合は偽。

式1 ｜｜ 式2 ｜｜ ・・・｜｜式n

式1から式nまですべてが偽なら偽。
それ以外の場合は真。

AND と OR が混在するような複雑な論理式を用いるときには、括弧をうまく用いて表現する。

19

3項関係の正しい書き方

間違い

×

a ＜ b ＜ c

a ＝ b ＞ c

数学での書き方は、
C言語ではできない。
（数学とは異なる意味で
実行される。）

NG

正しい

○

(a<b) && (b< c)　aがbより真に小さく、かつ
bがcより真に小さいとき 真。
それ以外のときは 偽。

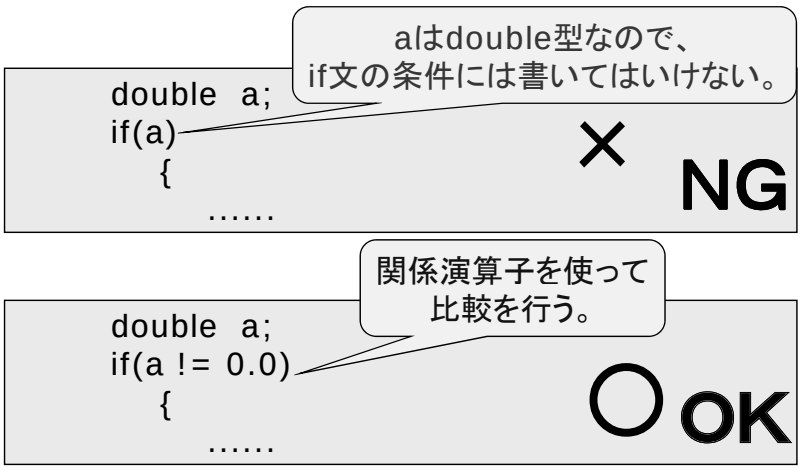
(a＝＝b) && (b＞ c)　aとbが等しく、かつ
bがcより真に大きいとき 真。
それ以外のときは 偽。

OK

20

論理値と型

論理値はint型で扱うこと。(スタイル規則参照)
したがって、論理演算子の被演算項はすべてint型にする。



21

練習3

```
/* logic.c      論理演算子実験( コメント省略 ) */
#include<stdio.h>
int  main()
{
       int a;
       int b;
       int c;
       printf("3つの整数を入力して下さい¥n");
       printf("a=");
       scanf("%d", &a);
       printf("b=");
       scanf("%d", &b);
       printf("c=");
       scanf("%d", &c);
/*  次のページに続く  */
```

22

```
/*続き*/
        if((a<b) && (a<c))
            {
                printf("aが最小です。¥n");
            }

        if((b<a) && (b<c))
           {
                printf("bが最小です。¥n");
           }

        if((c<a) && (c<b))
           {
                printf("cが最小です。¥n");
           }

        return 0;
}
```

23

多分岐（else-ifによる）

書式

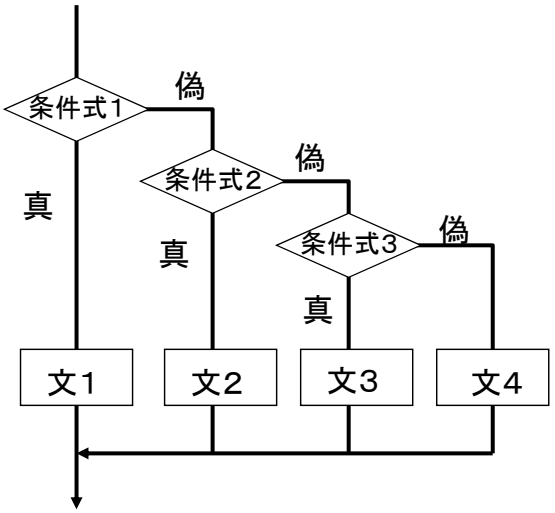
```
if(条件式1)
    {
        選択実行部分1
    }
else   if(条件式2)
    {
        選択実行部分2
    }
.
.
.
else  if(条件式n)
   {
        選択実行部分n
   }
else
  {
        選択実行部分(n+1)
  }
```

式は上から評価されて、
真になった条件式に対応する
選択実行部分が実行される。

すべての条件式が偽なら、
最後のelseの選択実行部分が
実行される。

24

多分岐のフローチャート




```
if(条件式1 )
    {
        文1
    }
else  if(条件式2)
    {
        文2
     }
else  if (条件式3)
    {
        文3
     }
else
    {
        文4
     }
```

25

多重分岐

選択実行部分中にも、if文を書く事ができる。

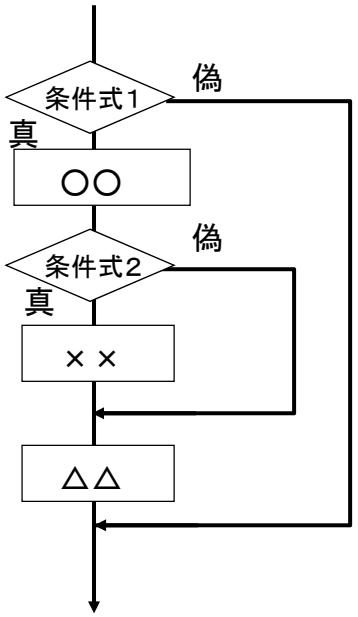
```
if(条件式)
    {
       選択実行部分1
    }
```

ここに、
またif文を書ける。

```
    if(a%2==1)
       {
          printf("aは奇数です。¥n");
          if(a>0)
             {
                printf("aは正の奇数です。¥n");
             }
       }
```

26

多重分岐のフローチャート




```
if(条件式1)
    {
        ○○
        if(条件式2)
            {
                ××
            }
        △△
    }
```

27

2次方程式を解くプログラム（p.67）

```
/*
        作成日:yyyy/mm/dd
        作成者:本荘太郎
        学籍番号:B00B0xx
        ソースファイル:quad_equation.c
        実行ファイル:quad_equation
        説明:
                2次方程式a(x*x)+bx+c=0の解を求めるプログラム。
                数学関数を用いるので、-lmのコンパイルオプションが必要。
        入力:
                標準入力から3つの係数a,b,cを入力する。
                aには0以外の任意の実数値を入力する。
                b,cには任意の実数値を入力する。
                a,b,cの順序に入力する。
        出力:
                標準出力に2つの解（実数値）を出力する。
*/
/*      次のページに続く        */
```

```
/* 続き */

#include <stdio.h>
#include <math.h>
int  main()
{
        /*      変数宣言                */
        double a;       /*2次の係数*/
        double b;       /*1次の係数*/
        doulbe c;       /*定数項*/

        double dis;     /*判別式(discriminant)の値*/
        double root_dis; /* 判別式の値の平方根*/

        double sol1;    /*小さい方の解*/
        double sol2;    /*大きい方の解*/

/*続く*/
```

29

```
/* 続き */
        /* 係数、定数項の入力*/
        printf("2次の項の係数を入力して下さい。a=？¥n");
        scanf("%lf", &a);
        printf("1次の項の係数を入力して下さい。b=？¥n");
        scanf("%lf", &b);
        printf("定数項を入力して下さい。c=？¥n");
        scanf("%lf", &c);

        /*入力値チェック*/
        if(a == 0.0)
          {
                /*a=0.0のときは、2次方程式でないので終了*/
                printf("2次の係数aは0.0以外にして下さい。¥n");
                return  -1;
          }
        /* これ以降では a は0.0以外*/

/*続く  */
```

30

```
/* 続き */
        dis = b*b - 4.0*a*c;        /* 判別式の計算 */

        if(dis>=0.0)  /* 実数解が存在する場合 */
           {
                /* disは正なので平方根を計算できる */
                root_dis = sqrt(dis);
                sol1=((-b)-root_dis)/(2.0*a);
                sol2=((-b)+root_dis)/(2.0*a);
                printf("(%6.2f)(x*x)+(%6.2f)x+(%6.2f)=0.0¥n"
                        , a, b, c);
                printf("の解は、%6.2fと%6.2fです。¥n", sol1, sol2);
           }
        else  /* 実数解が存在しない場合 */
           {
                printf("(%6.2f)(x*x)+(%6.2f)x+(%6.2f)=0.0¥n"
                        , a, b, c);
                printf("を満たす実数解はありません。¥n");
            }

        return 0;
}
```


実行例1

```
$ ./quad_equation
2次の項の係数を入力して下さい。a=？
1.0
1次の項の係数を入力して下さい。b=？
-3.0
定数項を入力して下さい。c=？
2.0
(  1.00)(x*x)+( -3.00)x+(  2.00)=0.0
の解は、   1.00 と   2.00です。
$
```

実行例2

```
$./quad_equation
2次の項の係数を入力して下さい。a=？
1.0
1次の項の係数を入力して下さい。b=？
1.0
定数項を入力して下さい。c=？
1.0
(  1.00)(x*x)+(  1.00)x+(  1.00)=0.0
を満たす実数解はありません。
$
```