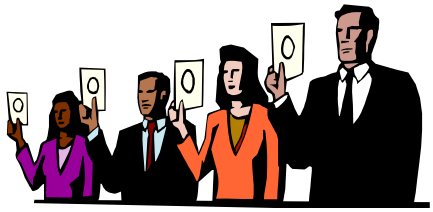
第8回　回数による繰り返し



1

今回の目標

- for文による繰り返し処理を理解する。
- 配列とfor文の組み合わせ方を理解する。
- 多重ループを理解する。

☆平均値を求めるプログラムを作る。

2

平均

n 個のデータ $x_0, x_1, \cdots, x_{n-1}$

の平均値

$$\overline{x} = \frac{x_0 + x_1 + \cdots + x_{n-1}}{n}$$

$$= \frac{\sum_{i=0}^{n-1} x_i}{n}$$

3

for文

式2（論理式）が真である間、命令を繰り返し実行する。

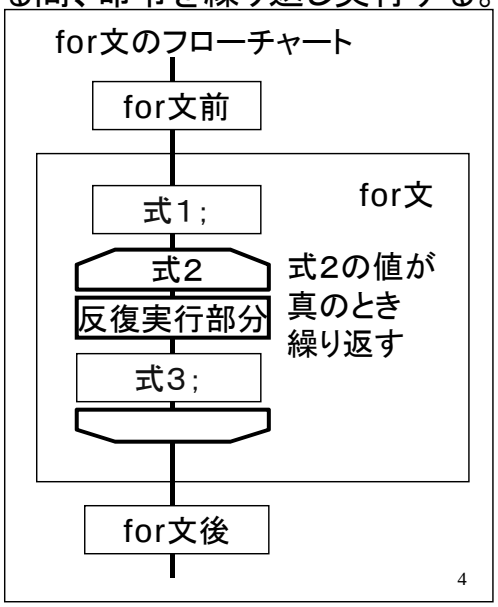
書式

```
for(式1;式2;式3)
    {
        反復実行部分
    }
```

式1：ループカウンタの初期化

式2：反復を続ける条件

式3：ループカウンタの
インクリメント



4

for文

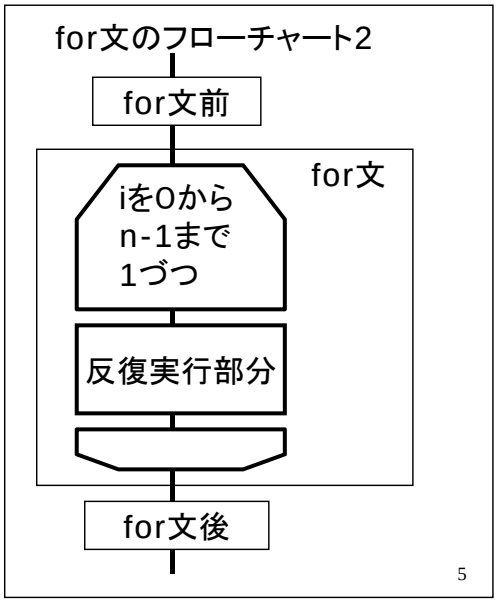
典型的な使い方

```
for(i=0;i<n;i++)
    {
        反復実行部分
    }
```

i=1：ループカウンタの初期化

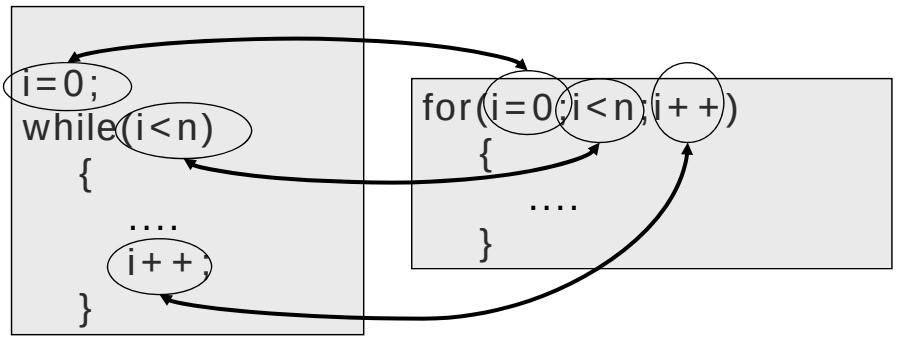
i<n：反復を続ける条件

i++：ループカウンタの
インクリメント



5

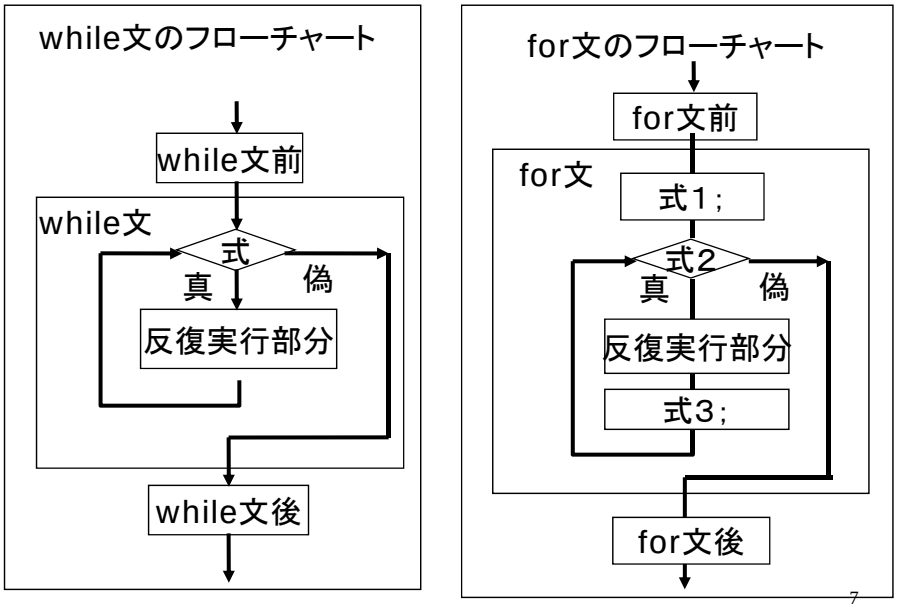
while 文との比較



for文では、回数指定の繰り返しの制御記述を、
一個所にまとめている。

6

while文とfor文のフローチャート



7

while文とfor文の書き換え

```
式1;
while(式2)
    {
        反復実行部分
        式3;
    }
```

⟺

```
for(式1;式2;式3)
    {
        反復実行部分
    }
```

回数で制御する繰り返しのときには、制御に必要な記述がfor文の方がコンパクトにまとまって理解しやすい。

逆に、繰り返しを論理値で制御するときは、while文で書くと良い。

8

練習1

```
/*for_test.c      回数指定反復実験( コメント省略 ) */
#include<stdio.h>
int  main()
{
        int n; /*反復回数用*/
        int  i; /*ループカウンタ*/

        printf("反復回数を入力して下さい¥n");
        scanf("%d",&n);
        for(i=0;i<n;i++)
          {
              printf("反復 %d回目 ¥n",i);
          }
        return  0;
}
```

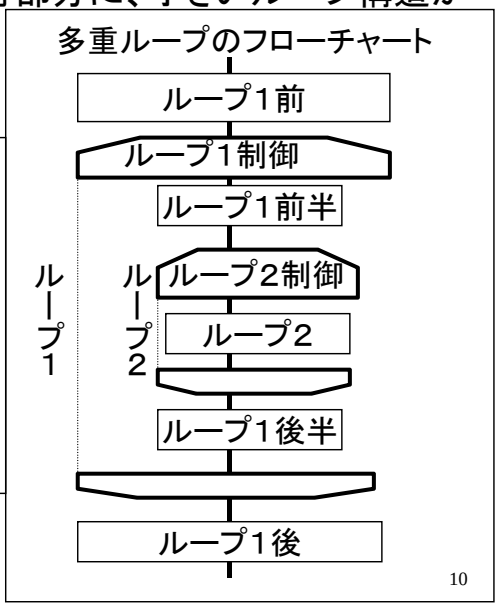
9

多重ループ

ループ構造の反復実行部分に、小さいループ構造が入っている制御構造。

典型的な例

```
for(式1;式2;式3)
    {
       ループ1前半
       for(式4;式5;式6)
          {
             ループ2
          }
       ループ1後半
    }
```

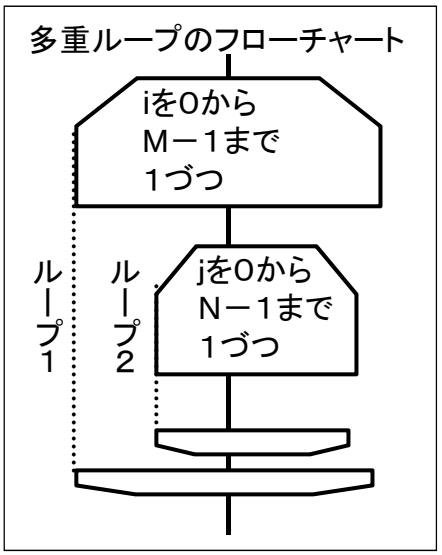


10

多重ループとループカウンタ

典型的な例

```
/*外側の反復回数*/
#define M 10
/*内側の反復回数*/
#define N 10
int  i; /*外側のカウンタ*/
int  j; /*内側のカウンタ*/
for(i=0;i<M;i++)
    {
      for(j=0;j<N;j++)
          {
              ループ2
          }
    }
```



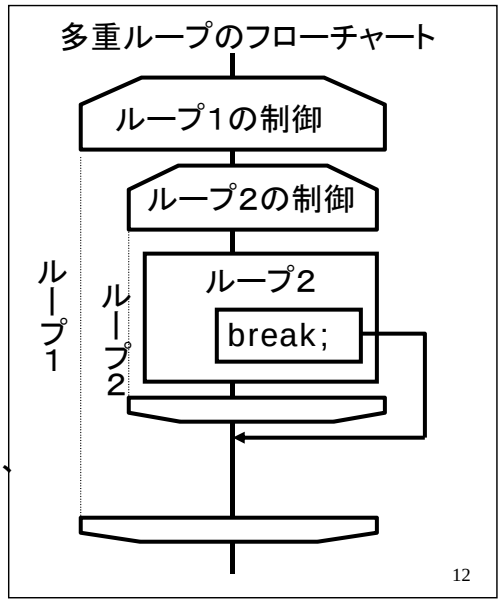
多重ループでは、ループ事に
別のループカウンタを用いる。

11

多重ループとbreak文

典型的な例

```
for(式1;式2;式3)
    {
      for(式4;式5;式6)
          {
              (break;)
          }
    }
```



注意:
多重ループ内のbreak文は、
一つだけ外側のループに
実行を移す。

12

練習2

```
/*多重ループ実験　multi_loop.c　　コメント省略　*/
#include<stdio.h>
int　main()
{
　　int i;　/* 外側のループカウンタ*/
　　int j;　/* 内側のループカウンタ*/

　　printf(" 九九を(半分だけ)表示¥n");

　　/*　次に続く　*/
```



```
　　for(i=1;i<10;i++)
　　{
　　　printf("%1dの段:",i);
　　　for(j=1;j<10;j++)
　　　{
　　　　printf("%1d * %1d　= %2d　",i,j,i*j);
　　　　if(i==j)
　　　　{
　　　　　break;
　　　　}
　　　}
　　　printf("¥n");
　　}
　　return 0;
}
```

平均値を求めるプログラム

```
/*
        作成日: yyyy/mm/dd
        作成者:本荘太郎
        学籍番号:B0zB0xx
        ソースファイル:average.c
        実行ファイル:average
        説明:n個の実数から平均値を求めるプログラム。
        入力:データ数として、
                標準入力からデータの個数を表す
                1つの正の整数nを入力。
                続いて、データとして、
                標準入力からn個の実数を任意の順番で入力。
        出力:標準出力にデータ中の最大値を出力。
*/
/*      次のページに続く      */
```

15

```
/*      続き    */
#include <stdio.h>
/*マクロ定義*/
#define    DATANUM     1000 /* データ個数の上限 */
int  main()
{       /*      変数宣言                */
        int     n;              /*  データ数*/
        int     i;      /* ループカウンタ、配列dataの添字 */
        double  ave;        /*平均値*/
        double  sum;        /*総和*/
        double  data[DATANUM];   /*データを入れる配列*/
        /* 入力処理*/
        /*データ数の入力*/
        printf("データ数を入力して下さい。¥n");
        printf("n= ?¥n");
        scanf("%d",&n);
/*      次のページに続く      */
```

```
/*データ数nのチェック*/
if(n<=0)
  {
     /*  不正な入力:データ数が0以下*/
     printf("データが無いのですね。¥n");
     return  -1;
  }
/* これ以降では、nは正の整数*/
if(n>DATANUM)
  {
     /* 不正な入力:上限オーバー*/
     printf("データが多すぎます。¥n");
     return  -1;
  }
/* これ以降では、nは1からDATANUMまでの整数*/
```

17

```
/*続き*/
/* 標準入力から配列へデータを読み込む/
for(i=0;i<n;i++)
   {
      /* n個の実数データの入力  */
      printf("data[%d] = ?",i);
      scanf("%lf",&data[i]);
   }

/*     次に続く       */
```

18

```
        /*初期設定*/
        sum=0.0;

        /*総和の計算*/
        for(i=0;i<n;i++)
           {
              sum=sum+data[i];
            }

        /*平均の計算　*/
        ave=sum/(double)n;

        /*出力処理*/
        printf("平均値は %6.2fです。¥n",ave);
        return 0;
}
```

実行例1

```
$make
gcc  average.c  -o  average
$ ./average
データ数を入力して下さい。
n= ？
3
data[0]= ？4.0
data[1]= ？2.0
data[2]= ？-3.0
平均値は  1.00 です。
$
```

実行例2

```
$./average
データ数を入力して下さい。
n= ？
0
データが無いのですね。
$
```

実行例3

```
$./average
データ数を入力して下さい。
n= ？
2000
データが多すぎます。
$
```