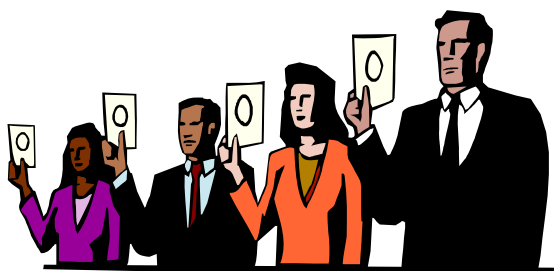


第7回 繰り返しⅡ （回数による繰り返し）



1

今回の目標

- for文による繰り返し処理を理解する。
- 多重ループを理解する。

☆等差数列を計算するプログラムを作る。

2

等差数列

初項が a 、公差が d の
等差数列 a_i ($i = 0, 1, 2, \dots, n$)
を第 n 項まで計算する。

$$a_0 = a, \quad a_1 = a_0 + d, \quad a_2 = a_1 + d, \\ \dots, \quad a_n = a_{n-1} + d$$

一般項 a_i は以下のような漸化式をみたす。

$$a_i = a_{i-1} + d$$

3

for文

条件式(論理式)が真である間、命令を繰り返し実行する。

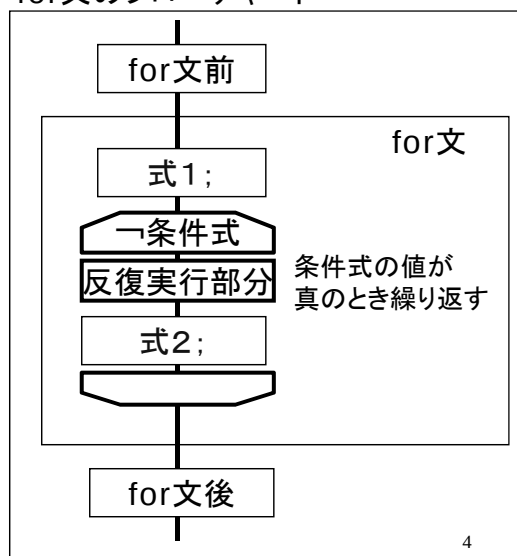
書式

```
for(式1;条件式;式2)
{
    反復実行部分
}
```

式1 : ループカウンタの
初期化
条件式: 反復を続ける条件
(偽になったら終了)
式2 : ループカウンタの
インクリメント

for文は
ループカウンタを
一個所に記述できる。

for文のフローチャート



4

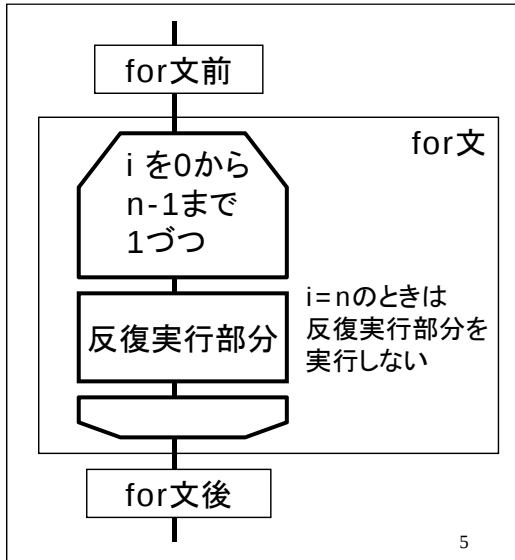
for文

典型的な使い方

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

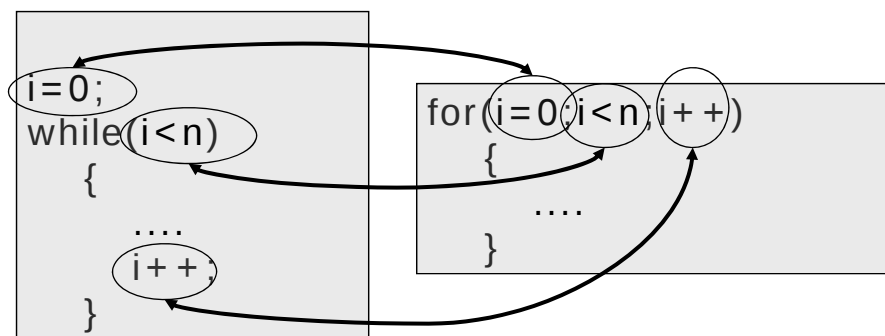
i=0 : ループカウンタ *i* の初期化
i<*n* : 反復を続ける条件
 (*i* が 0, 1, ..., *n*-1 のとき)
i++ : ループカウンタ *i* の
 インクリメント

*n*回繰り返しの定番
 パターンとして記憶する
 と良い。
 不等号に注意。

for文のフローチャート
(この演習での省略記法)

5

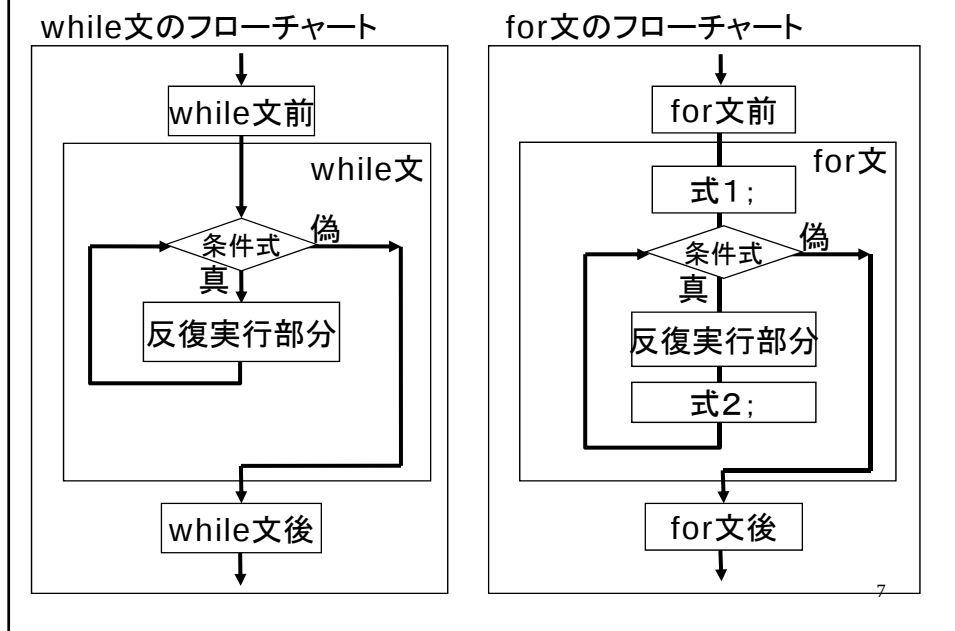
while 文との比較



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



while文とfor文の書き換え

```

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



```

for(式1;条件式;式2)
{
    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。

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 */
    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。

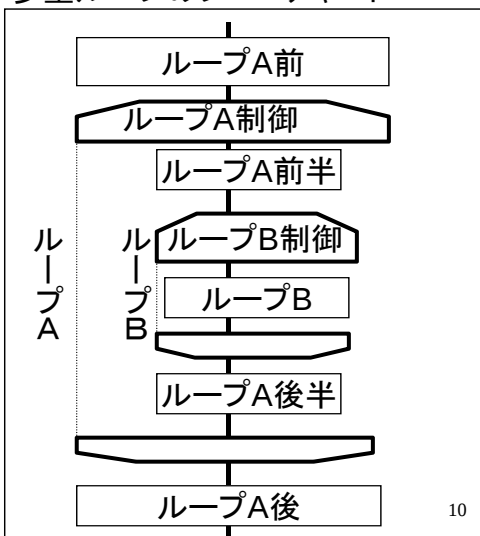
多重ループのフローチャート

典型的な例

```

for(式A1; 条件式A; 式A2)
{
    ループA前半
    for(式B1; 条件式B; 式B2)
    {
        ループB
    }
    ループA後半
}

```



10

多重ループとループカウンタ

典型的な例

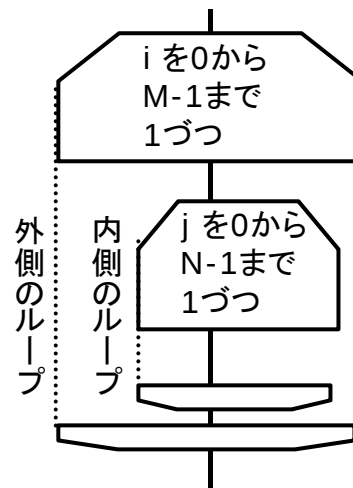
```
#define M 10 /*外側の反復回数*/
#define N 10 /*内側の反復回数*/

int i; /*外側のループのカウンタ*/
int j; /*内側のループのカウンタ*/

for(i=0; i<M; i++)
{
    /* 外側のループ */
    for(j=0; j<N; j++)
    {
        /* 内側のループ */
    }
}
```

多重ループでは、ループごとに別のループカウンタを用いる。

多重ループのフローチャート



11

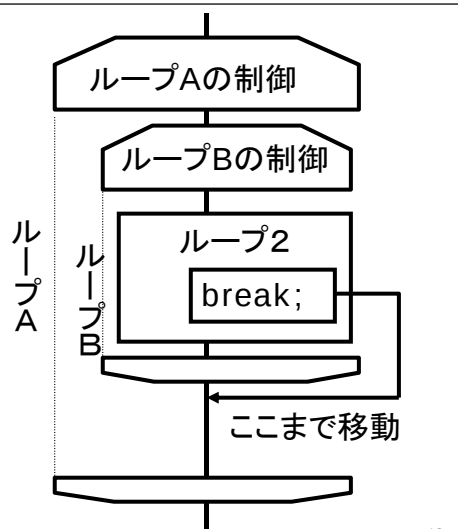
多重ループとbreak文

典型的な例

```
for(式A1;条件式A;式A2)
{
    for(式B1;条件式B;式B2)
    {
        ( 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。

多重ループのフローチャート



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");

    /* 次に行く */
}
```

13

```
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;
}
```

14

等差数列を表示するプログラム

```
/*
    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: sequence.c
    実行ファイル: sequence
    説明: 初項a、公差dの等差数列を
          第n項まで表示するプログラム。
          ただし、初項は第0項であるとする。
    入力: 初項a、公差d、最終項の番号nをこの順に
          標準入力から入力する。
          初項と公差は任意の実数、
          最終項の番号は正の整数とする。
    出力: 標準出力に数列の各項の値を出力する。
          数列の各項の値は実数である。
*/
/*  次のページに続く  */
```

15

```
/*  続き  */
#include <stdio.h>

int main()
{
    /* 変数宣言 */
    double seq;          /* 数列の各項 */
    double first;         /* 初項 */
    double difference;    /* 公差 */
    int n;                /* 項数 */
    int i;                /* ループカウンタ */

    /* 初項の入力 */
    printf("初項aの値を入力して下さい。¥n");
    printf("a= ?¥n");
    scanf("%lf", &first);
    /*  次のページに続く  */
```

```
/* 続き */
/* 公差の入力 */
printf("公差を入力して下さい。¥n");
printf("d= ?¥n");
scanf("%lf", &difference);
/* 項数の入力 */
printf("項数を入力して下さい。¥n");
printf("n=?¥n");
scanf("%d",&n);

/* 入力データのチェック */
if (n<=0){
    /* 不正な入力:項数が0以下 */
    printf("項数は正の整数で与えてください¥n");
    return -1;
}
/* 次のページに続く */
```

```
/* 続き */
/* 数列の計算と出力 */
/* 初期設定(初項) */
seq=first;
printf("a%2d: %6.3f¥n", 0, seq);

/* 繰り返し処理 */
for(i=0;i<n;i++)
{
    seq=seq+difference;
    printf("a%2d: %6.3f¥n", i+1, seq);
}

return 0;
}
```

実行例1

```
$make
gcc sequence.c -o sequence
$ ./sequence
初項を入力して下さい
a= ?
-7
公差を入力して下さい
d= ?
3
項数を入力して下さい
n= ?
2
a0: -7.000
a1: -4.000
a2: -1.000
$
```

実行例2

```
$. /sequence
初項を入力して下さい
a= ?
-7
公差を入力して下さい
d= ?
3
項数を入力して下さい
n= ?
-2
項数は正の整数で与えてください
$
```