

すべてができる！？

これまで使ってきた命令を使えば、標準入力(コンソール画面のキー入力)で得たデータを加工(計算などの各種の処理)し、結果として標準出力(コンソール画面に出す)するプログラムは、**どんなものでも作る**ことができると言って過言ではありません。

しかし、もっと規則を知っていないと、非常に長いプログラムを必要とする場合があります。もっと知っていると、分かりやすく、簡単書ける場合がたくさんあります。これより、そのような規則を紹介して行きます。

これまで行ってきた後判定の do while と、前判定の while

後ろ判定の while をこれまで行ってきましたが、前判定の while も存在します。

以下で比較します。これまで行ってきた**後判定の do while**を示します。

後判定は、繰り返し対象を条件に関係なく1回行ってから、続けて繰り返すか判定します。

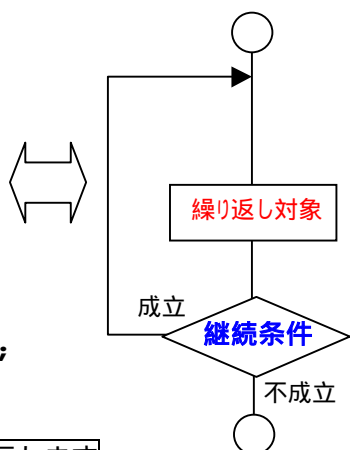
繰り返しの中のオペレータ入力で

繰り返しを判定する場合などで

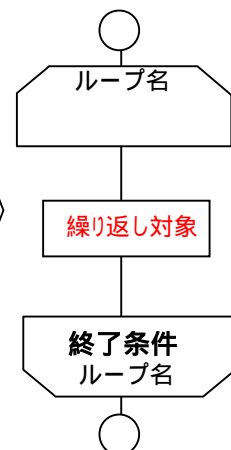
使われます。

```
do {
    繰り返し対象の処理
} while( 継続条件 );
```

事後判定繰り返し形



ループ端を利用した事後判定繰り返し形

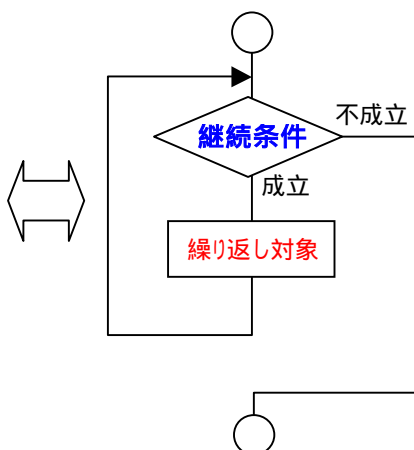


対して前判定の while を示します

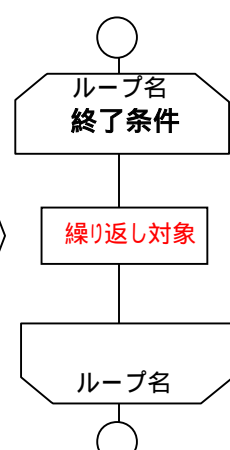
大きな違いは、始めに繰り返すかどうかを、判定して不成立の場合は、一回も繰り返しをしないとことです。

事前判定繰り返し形

```
while( 継続条件 ){
    繰り返し対象の処理
}
```



ループ端を利用した事前判定繰り返し形



簡単な例で、入力した値の個数の を表示させたい場合のプログラムを示します。

実行例

個数>3

個数>6

個数>0

```
#include <stdio.h>
main()
{
    int kai;          /*個数入力用 */
    int n = 0;        /* 数える変数 */

    printf("個数>");
    scanf("%d" , & kai );
    if( 1 ){
        do {
            printf(" ");
            n = n + 1; /* 数える。 */
        } while (n < kai);
    }
    printf("¥n");
}
```

問題1 左のプログラムを作成 (tA01.c) して、実験して見ましょう。

左のプログラムでは、ifの条件が、1の成立なので必ず、doを実行する。つまり、ifが存在しないのと同じことになっている。0回数の入力で、正しくないことを確認して ifの条件を正しく直さない。

答えは、(n < kai)です

。このような場合に、次のような前判定の while を使うと簡単になるのです。

```
#include <stdio.h>
main()
{
    int kai; /*個数入力用 */
    int n; /* 数える変数 */
    printf("個数>");
    scanf("%d" , & kai );

    n = 0;
    while (n < kai){
        printf(" ");
        n = n + 1; /* 数える。 */
    }
    printf("¥n");
}
```

問題2 左のプログラムを作成 (tA02.c) して、確認しましょう。次にこのプログラムを変更します。

の一つが10の大きさとして、500までの入力を行わせませす。つまり30の入力で、3個並べばよいのです。なお、35の入力なら4個並び、34の入力なら3個並ぶように、(入力個数/10)の四捨五入した個数だけ並ぶように変更しましょう。(余りの%演算子を使うと簡単です。使わなくてもできますが…)

問題3 前述の tA02.c をコピーして、tA03c を作り次のように、個数を減らしながら1個まで並べるプログラムに変更しなさい。

個数>35

個数>34

個数>47

問題3の 解答例

```

#include <stdio.h>
main()
{
    int kai;      /* 回数入力用 */
    int n = 0;    /* 数える変数 */
    printf("回数>");
    scanf("%d" , &kai );

    if(kai % 10 >= 5){
        kai = kai / 10 + 1;
    } else {
        kai = kai / 10;
    }

    while(kai > 0){
        n = 0;
        while (n < kai){
            printf(" ");
            n = n + 1;    /* 数える。 */
        }
        printf("¥n");
        kai = kai - 1;
    }
}

```

```

kai = (kai+5) / 10;

```

⇔

どちらでもよいが、上の方が、比較的簡単で、応用しやすいでしょう。

```

#include <stdio.h>
main()
{
    int kai;      /* 回数入力用 */
    int n = 0;    /* 数える変数 */
    printf("回数>");
    scanf("%d" , &kai );

    while(kai >= 5){
        n = 0;
        while (n + 5 <= kai){
            printf(" ");
            n = n + 10;    /* 数える。 */
        }
        printf("¥n");
        kai = kai - 10;
    }
}

```