

第4章 「プログラムの流れの繰り返し」の まとめ

4.1 用語のまとめ

教 p.60

do ~ while 文

do 文 while (式) ;

まず、文(_____ と呼ばれる)を実行する。式が偽(0)にならない限り、文の実行を _____。

注: 必ず 1 回はループ本体を実行する。

教 p.61

複合文(ブロック)内での宣言 ブロックの中で宣言された変数は、その _____ である。

教 p.64

初期化子 変数は次の形で初期化することができる。

型名 変数名₁ = 初期化子₁, ... , 変数名_n = 初期化子_n;

初期化子 (initializer) は今のところ “式” (expression) と思っておいて良い。

教 p.66

複合代入演算子 *=, /=, %=, +=, -=, などのことである。例えば、sum += no は _____ と同じ意味になる。

後置増分演算子・前置増分演算子

教 p.67

a++ a の値を一つだけ増やす (式全体の値は、_____)

教 p.73

a-- a の値を一つだけ減らす (式全体の値は、_____)

++a a の値を一つだけ増やす (式全体の値は、_____)

--a a の値を一つだけ減らす (式全体の値は、_____)

教 p.68

while 文

while (式) 文

_____ が偽(0)でない限り、文(ループ本体)を _____。

注: ループ本体が一度も実行されないことがある。

do ~ while 文

while 文

教 p.69

文字定数 文字を _____ で囲んだもののことである。¥n や ¥t などの拡張表記は 1 文字として扱われる。

教 p.74

for 文

for (式₁; 式₂; 式₃) 文

ループに入る前にまず _____ を実行する。

_____ が真 (非 0) である間、_____ を繰り返し実行する。

詳細: 式₁ ~ 式₃ は省略可能である。式₂ を省略したときは、1 (真) と書くのと同じ意味になる。

for 文

左のような for 文は右に示す while 文と (ほぼ) 等価である。

for 文	(ほぼ) 等価な while 文
<pre> for (A ; B ; C) { ループ本体 } </pre>	<pre> _____ _____ _____ _____ _____ </pre>

一定回数の繰返し for 文には、良く使う決まり文句的な形がある。

教 p.76

```

for ( i = 0; i < n; i++) ...   i が _____ n 回繰り返す
for ( i = 1; i <= n; i++) ... i が _____ n 回繰り返す
for ( i = n; i > 0; i--) ...  i が _____ n 回繰り返す
for ( i = n-1; i >= 0; i--) ... i が _____ n 回繰り返す

```

多重ループ for 文や while 文などのループ本体が、また for 文や while 文などの繰返し文になっていることを二重ループという。二重・三重・… ループをまとめて、多重ループという。

教 p.78

キーワード if や else など C 言語にとって特別な意味のある単語を _____ (keyword) と呼ぶ。変数名などに使用することはできない。(ただし、変数名の一部に使用するの構わない。)

教 p.82

自由形式 C 言語では、原則としてレイアウト (空白の数や改行) はプログラムの意味に影響を及ぼさない。空白がいくつ連続しても空白 1 文字と同じであり、改行も空白と同じである。

教 p.84

注意すべきところ:

- 前処理指令 (_____, #define ...) などの途中では、改行できない。
- 文字列定数、文字定数の途中でも、改行できない。

4.2 プログラム例

整数値を逆順に

```

1  #include <stdio.h>
2
3  int main(void) {
4      int num = 12345;
5
6      do {
7          printf("%d", num % 10);
8          num = num / 10;
9      } while (num > 0);
10
11     return 0;
12 }

```

num の値の変化

	1 回目	2 回目	3 回目	4 回目	5 回目
7 行	_____				
9 行	_____				

典型的な for 文 (階乗の計算)

```

1  #include <stdio.h>
2
3  int main(void) {
4      int i, n, fact = 1;
5      printf("正の数を入力してください。 ");
6      scanf("%d", &n);
7
8      for (i = 1; /* 通常 1行に書くところを 3行にわけた。*/
9             i <= n;
10            i++) {
11          fact = fact * i;
12        }
13        printf("あなたの入力した数の階乗は %dです。 \n", fact);
14
15        return 0;
16    }

```

i, fact の値の変化 (n が 5 のとき)

行	1 回目		2 回目		3 回目		4 回目		5 回目	
	i	fact	i	fact	i	fact	i	fact	i	fact
9	—	—	—	—	—	—	—	—	—	—
11	—	_____	—	_____	—	_____	—	_____	—	_____
10	_____	—	_____	—	_____	—	_____	—	_____	—

典型的な for 文 (正 n 角形の座標の出力)

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main(void) {
5     int n, i;
6
7     printf("nを入力して下さい: "); scanf("%d", &n);
8
9     for(i = 0; i < n; i++) {
10        double theta1 = 2 * M_PI * i / n;
11        double theta2 = 2 * M_PI * (i+1) / n;
12        printf("%.3f %.3f %.3f %.3f\n",
13            100 * cos(theta1), 100 * sin(theta1),
14            100 * cos(theta2), 100 * sin(theta2));
15    }
16
17    return 0;
18 }
```

二重ループ (数の三角形)

```
1 #include <stdio.h>
2
3 int main(void) {
4     int i, j, n;
5     printf("nを入力して下さい:"); scanf("%d", &n);
6     for (i=1; i<=n; i++) {
7         for (j=1; j<=i; j++) {
8             printf("%d", j%10);
9         }
10        printf("\n");
11    }
12    return 0;
13 }
```

1
12
123
1234

二重ループ (ダイヤモンド 図形の座標の出力)

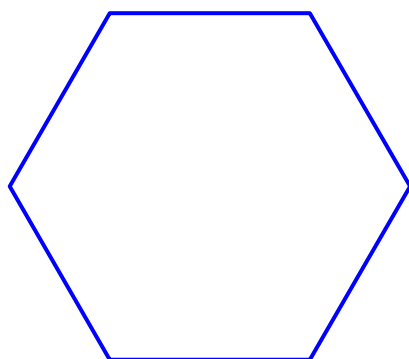
```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main(void) {
5     int n, i, j;
6
7     printf("nを入力して下さい: "); scanf("%d", &n);
8
9     for (i = 0; i < n; i++) {
10        double theta1 = 2 * M_PI * i / n;
11        for (j = i+1; j < n; j++) {
```

```

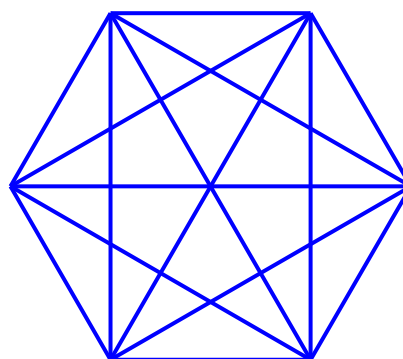
12         double theta2 = 2 * M_PI * j / n;
13         printf("%.3f %.3f %.3f %.3f\n",
14                100 * cos(theta1), 100 * sin(theta1),
15                100 * cos(theta2), 100 * sin(theta2));
16     }
17 }
18
19 return 0;
20 }

```

正 n 角形 (n=6)



ダイヤモンド図形 (n=6)



4.3 文法のまとめ

文 (statement) に以下を追加

分類	一般形	補足説明
do~while 文	do 文 while (式);	(教科書 p.60)
while 文	while (式) 文	(教科書 p.68)
for 文	for (式; 式; 式) 文	(教科書 p.74)
continue 文	continue ;	詳細な説明は割愛

式 (expression) に以下を追加する。

分類	一般形	補足説明
後置演算	式 後置演算子	Cの後置演算子は ++, --のみ (教科書 p.66)