特設

第5章 スコープ（利用可能な範囲），寿命，初期値の取り扱い…違いをきちんと理解する

「局所（ローカル）変数」と「大域（グローバル）変数」

鹿取 祐二

リスト1　局所変数のスコープ…有効範囲は宣言された関数内部に限られる

```
void main(void)
{
int a, b;            // main関数内でのみ利用可能

}

void func(void)
{
    a = b + 4;       // aとbの利用は文法違反
}
```

（a）他の関数内に宣言されている変数を利用しようとすると文法違反

```
void main(void)
{
int a, b;

}

void func(void)
{
int a, b;          // main関数内と同じ識別子でも良い
                   // main関数内のa,bとは異なるもの
}
```

（b）異なる関数内で同じ識別子の変数を宣言しても問題ない

複数の関数を作成し始めると，変数宣言に関して幾つかの疑問が出てきます．

例えば，異なる関数間で同じ名前の変数を宣言しても大丈夫なのか，他の関数内に宣言されている変数を別の関数から操作できるのか，などの疑問です．本章では，変数宣言に関する文法を紹介します．

● 変数宣言は2つに分類される

C言語による変数宣言は，局所変数と大域変数の2つに分類されます．局所変数とは，関数の内部に宣言された変数であり，ローカル変数と呼ぶこともあります．一方，関数の外側に宣言された変数を大域変数とかグローバル変数と呼びます．両者の見かけ上の記述方法は，ほぼ同じと考えて問題ありません（記述できる初期値に多少の差がある）．

ただし，両者の変数としての性質は全く異なります．特にスコープ（利用可能な範囲），寿命，初期値の取り扱いが異なります．次に順番に両者の違いを説明します．

違い①…変数の有効範囲

● 局所変数の場合

局所変数と大域変数はスコープが異なります．スコープとは，変数の識別子（名前）の有効範囲のことです．

局所変数（仮引数を含む）の場合，識別子の有効範囲は宣言された関数の内部に限られます．従って，**リスト1**（a）のように他の関数内に宣言されている変数を利用しようとすると文法違反になります．

リスト1（b）のように異なる関数内で同じ識別子の変数を宣言しても問題はありません．両者には，異なる記憶領域が割り当てられているからです．

● 大域変数の場合

大域変数の識別子は，全ての関数から共通に利用できます．**リスト2**のように，異なるソース・ファイルに宣言された大域変数であっても，externの記憶クラスを使って他のソース・ファイルで本体を定義していることを宣言すれば利用できます．

違い②…変数の寿命（利用可能な期間）

局所変数と大域変数では，変数としての寿命，すなわち，いつ使用可能になって，いつ使用不可能になるのかが異なります（**リスト3**）．

● 局所変数の場合

局所変数は関数内部だけのものなので，宣言のある関数が呼ばれた時点で作成され（記憶場所が確保され），使用可能になります．また，目的の関数が終了すると記憶領域が解放され，使用できなくなります．つまり，目的の関数が実行中のときのみ存在します．