C/C++プログラミング

第5回

配列と多次元配列の知識と演習

配列: C++

List5-2 配列

```
----------------------------------------
// 配列の各要素に1, 2, 3, 4, 5を代入して表示

#include <iostream>

using namespace std;

int main()
{
        int a[5];                       // 要素型がint型で要素数5の配列

        a[0] = 1;
        a[1] = 2;
        a[2] = 3;
        a[3] = 4;
        a[4] = 5;

        cout << "a[" << 0 << "] = " << a[0] << '\n';
        cout << "a[" << 1 << "] = " << a[1] << '\n';
        cout << "a[" << 2 << "] = " << a[2] << '\n';
        cout << "a[" << 3 << "] = " << a[3] << '\n';
        cout << "a[" << 4 << "] = " << a[4] << '\n';
}
----------------------------------------
```

配列: C++

「int a[5]; 」
整数型の配列 a[5] を宣言．配列とは，同じ変数名（この例では「a」）の付いた複数のデータをひとまとめにしたもの．「[5]」は5個のデータを格納できるという意味.

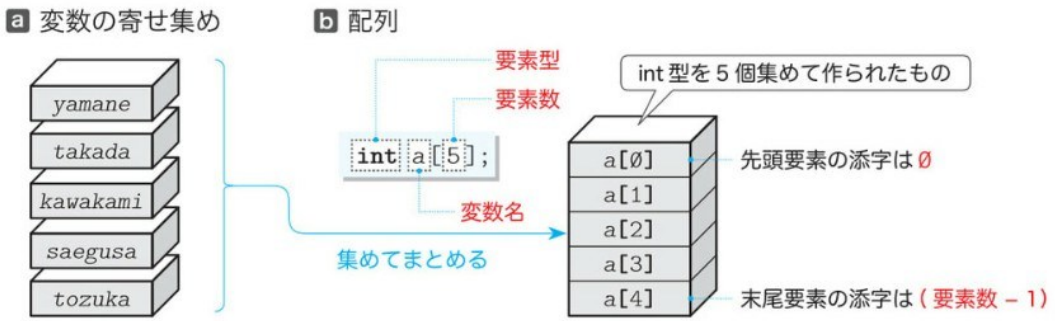


Fig.5-1 ばらばらに定義された変数と配列

「a[0] = 1;」
配列の中の一つ一つの格納場所（変数）は「a[0]」などとして指定する．角括弧の中の数字は「添字」と呼ばれる各格納場所につけられた番号．0から始まり，この場合４までである．a[0]=1; は配列の0番目の変数a[0]に1を代入せよという意味.

「cout << "a[" << 0 << "] = " << a[0] << '\n';」
配列の中の変数のデータは a[0]，a[1]，… などとして指定する.

以降，C++とCでは配列の扱いには大きな違いはないので，Cは省略.

配列: C++

List5-3 配列

```
----------------------------------------
// 配列の各要素に1, 2, 3, 4, 5を代入して表示（for文）

#include <iostream>

using namespace std;

int main()
{
        int a[5];                       // 要素型がint型で要素数5の配列

        for (int i = 0; i < 5; i++)
                a[i] = i + 1;

        for (int i = 0; i < 5; i++)
                cout << "a[" << i << "] = " << a[i] << '\n';
}
----------------------------------------
```

List5-2で類似の処理を続けている箇所を，for文を用いてシンプルにしたプログラム．プログラミングの簡素化は
「ソースコードを短くする→見やすくなり誤り(バグ)が少なくなる」
という効果があり極めて重要.

配列: C++

「
```
a[i] = i + 1;
```
」
「
```
cout << "a[" << i << "] = " << a[i] << '\n';
```
」
配列の添字は変数で表すことができる．上の例の場合 i が変数であり，i= 1 なら a[i] は a[1]，i = 2 なら a[i] は a[2] となる．

配列: C++

List5-4 配列の初期化

```
----------------------------------------
// 配列の各要素を1, 2, 3, 4, 5で初期化して表示

#include <iostream>

using namespace std;

int main()
{
        int a[5] = {1, 2, 3, 4, 5}; // 要素型がint型で要素数5の配列

        for (int i = 0; i < 5; i++)
                cout << "a[" << i << "] = " << a[i] << '\n';
}
----------------------------------------
```

「

```
int a[5] = {1, 2, 3, 4, 5};
```

」

配列の宣言文での初期化．a[0]=1, a[1]=2, … と格納されることを意味する．

配列: C++

List5-5 配列の要素数

```
// 配列の各要素を1, 2, 3, 4, 5で初期化して表示（要素数を計算によって求める）

#include <iostream>

using namespace std;

int main()
{
        int a[] = {1, 2, 3, 4, 5};
        int a_size = sizeof(a) / sizeof(a[0]); // 配列aの要素数

        for (int i = 0; i < a_size; i++)
                cout << "a[" << i << "] = " << a[i] << '\n';
}
```

「

```
int a[] = {1, 2, 3, 4, 5};
int a_size = sizeof(a) / sizeof(a[0]);
```

」

1行目は配列の要素数を指定せず宣言＆初期化．2行目は1行目で初期化された配列の要素数を計算し，a_size に代入している．sizeof(a)は配列aの格納場所全体の大きさ（バイト数）．sizeof(a[0])は変数 a[0] の大きさ（バイト数）．他の変数a[1], a[2], … の大きさも同じなので，sizeof(a)/sizeof(a[0]) の計算で配列aの中の変数の個数が計算できる．

配列: C++

List5-6 標準入力から配列への数値の代入

```
----------------------------------------
// ５人の点数を読み込んで合計点・平均点を表示

#include <iostream>

using namespace std;

int main()
{
        int tensu[5];                   // ５人の点数
        int sum = 0;                    // 合計

        cout << "5人の点数の合計点と平均点を求めます。\n";
        for (int i = 0; i < 5; i++) {
                cout << i + 1 << "番の点数：";
                cin >> tensu[i];                // tensu[i]を読み込んで
                sum += tensu[i];                // sumにtensu[i]を加える
        }

        cout << "合計は" << sum << "点です。\n";
        cout << "平均は" << static_cast<double>(sum) / 5 << "点です。\n";
}
----------------------------------------
```

配列: C++

「
```
cin >> tensu[i];
sum += tensu[i];
```
」
1行目は配列の変数に値を代入．2行目はその変数の値を sum に加算する．

配列: C++

List5-8 配列のデータの並びの反転

```
----------------------------------------
// 配列の要素の並びを反転して表示

#include <ctime>
#include <cstdlib>
#include <iostream>

using namespace std;

int main()
{
        const int n = 7;                    // 配列aの要素数
        int a[n];

        srand(time(NULL));                          // 乱数の種を初期化
        for (int i = 0; i < n; i++) {
                a[i] = rand() % 100;
                cout << "a[" << i << "] = " << a[i] << '\n';
        }

        for (int i = 0; i < n / 2; i++) {
                int t = a[i];
                a[i] = a[n - i - 1];
                a[n - i - 1] = t;
        }

        cout << "要素の並びを反転しました。\n";
        for (int i = 0; i < n; i++)
                cout << "a[" << i << "] = " << a[i] << '\n';
}
----------------------------------------
```

配列: C++

「

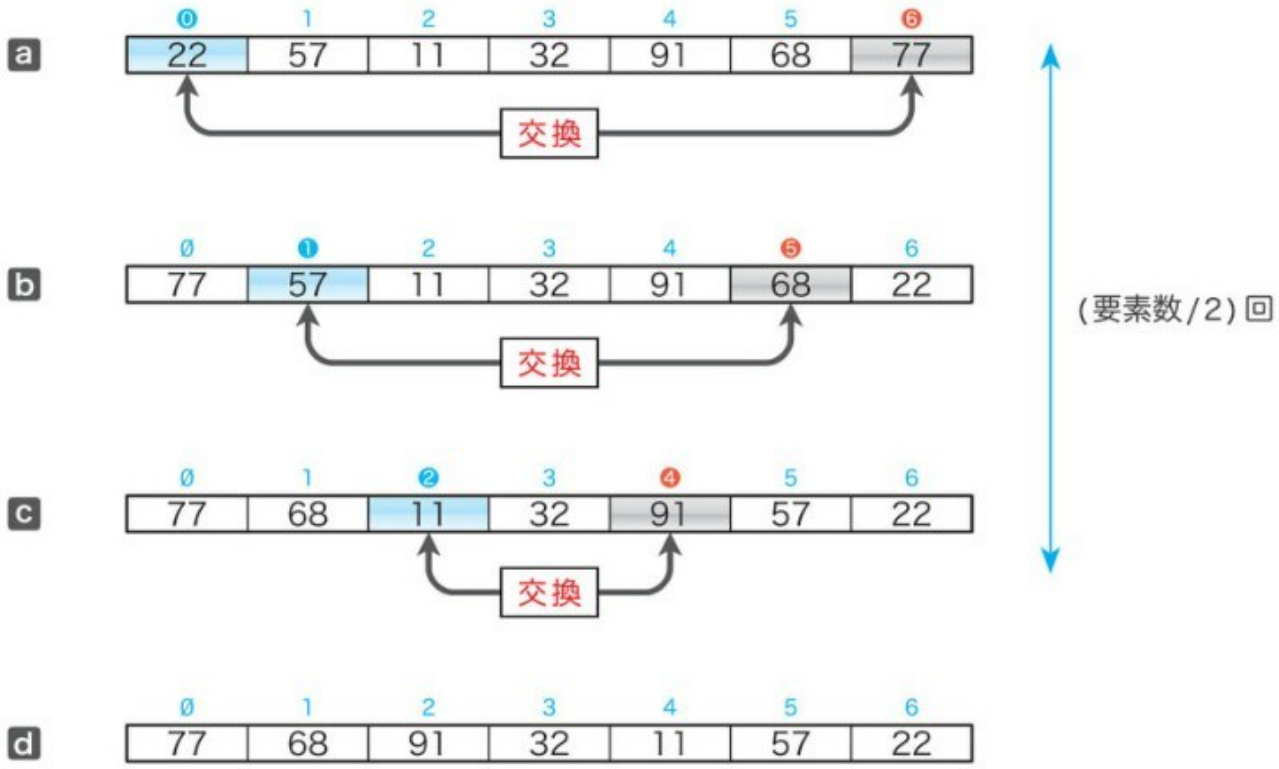
```
for (int i = 0; i < n / 2; i++) {
    int t = a[i];
    a[i] = a[n - i - 1];
    a[n - i - 1] = t;
}
```

」

a[i] と a[n – i – 1] の値の入れ替えを行っている.

- まずint t = a[i]; は一旦 a[i] の値を変数 t に格納して退避.
- その次の a[i] = a[n - i - 1] は退避した格納場所 a[i] に a[n – i – 1] の値を上書きコピー.
- 最後の a[n – i – 1] = t は上で退避しておいた t の値を a[n – i – 1] にコピーする.
- 以上で元々の a[i] と a[n – i – 1] の値の入れ替えが完了する.
- これを i=0 から i<n/2 まで繰り返す.

配列: C++



本書の図では、配列の要素を縦方向に並べたり横方向に並べたりします。

- 要素を縦に並べる場合は添字の小さい要素を上側にして、
- 要素を横に並べる場合は添字の小さい要素を左側にします。

Fig.5-4 配列の要素の並びを反転する

多次元配列: C++

List5-10 多次元配列（2次元配列の場合）

```
----------------------------------------
// 3行2列の２次元配列の全構成要素の値を読み込んで表示

#include <iostream>

using namespace std;

int main()
{
        int m[3][2];    // 3行2列の２次元配列

        cout << "各構成要素の値を代入せよ。\n";
        for (int i = 0; i < 3; i++) {
                for (int j = 0; j < 2; j++) {
                        cout << "m[" << i << "][" << j << "] : ";
                        cin >> m[i][j];
                }
        }

        for (int i = 0; i < 3; i++) {
                for (int j = 0; j < 2; j++) {
                        cout << "m[" << i << "][" << j << "]:" << m[i][j] << '\n';
                }
        }
}
----------------------------------------
```

多次元配列: C++

「
```
int m[3][2];
```
」
int型で3行2列の配列mを宣言.

「
```
for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 2; j++) {
        cout << "m[" << i << "][" << j << "] : ";
        cin >> m[i][j];
    }
}
```
」
多次元配列への値の代入や呼び出しには「入れ子」になったfor文を使うと良い．外側のfor文でiを更新しながら配列mの「行」を変えていく．内側のfor文では j を更新しながら配列mの「列」を変えていく.

「
```
cout << "m[" << i << "][" << j << "] : " << m[i][j] << '\n';
```
」
配列の各変数 m[i][j] の値の取り出し.

多次元配列: C++

List5-11 行列の加算

```
----------------------------------------
// 2行3列の行列を加算する

#include <iomanip>
#include <iostream>

using namespace std;

int main()
{
        int a[2][3] = { {1, 2, 3}, {4, 5, 6} };
        int b[2][3] = { {6, 3, 4}, {5, 1, 2} };
        int c[2][3];

        for (int i = 0; i < 2; i++)
                for (int j = 0; j < 3; j++)
                        c[i][j] = a[i][j] + b[i][j];

        cout << "行列a\n";                                          // 行列aの要素の値を表示
        for (int i = 0; i < 2; i++) {
                for (int j = 0; j < 3; j++)
                        cout << setw(3) << a[i][j];
                cout << '\n';
        }

        cout << "行列b\n";                                          // 行列bの要素の値を表示
        for (int i = 0; i < 2; i++) {
                for (int j = 0; j < 3; j++)
                        cout << setw(3) << b[i][j];
                cout << '\n';
        }

        cout << "行列c\n";                                          // 行列cの要素の値を表示
        for (int i = 0; i < 2; i++) {
                for (int j = 0; j < 3; j++)
                        cout << setw(3) << c[i][j];
                cout << '\n';
        }
}
----------------------------------------
```

多次元配列: C++

「

```
int a[2][3] = { {1, 2, 3}, {4, 5, 6} };
int b[2][3] = { {6, 3, 4}, {5, 1, 2} };
```

」

「

```
        for (int i = 0; i < 2; i++)
                for (int j = 0; j < 3; j++)
                        c[i][j] = a[i][j] + b[i][j];
```

」

このコードは、2×3 の二次元配列について
外側の i（0〜1）と内側の j（0〜2）を回す二重ループで、対応する要素を要素ごとに加算しています。

- for (int i = 0; i < 2; i++)：行を走査（0 行目、1 行目）
- for (int j = 0; j < 3; j++)：列を走査（0〜2 列目）
- c[i][j] = a[i][j] + b[i][j];：c の各要素に a と b の同じ位置の値を足す

要するに、c = a + b を要素ごとに計算している処理です。

課題

課題5-1 （**プログラム名：ex05_1.cpp**）
List5-3 を修正し，a[0]=5, a[1]=4, a[2]=3, … と逆順に入力し，それらの変数を表示するプログラムにする.

【実行例】

出力:

```
a[0] = 5
a[1] = 4
a[2] = 3
a[3] = 2
a[4] = 1
```

課題5-2（**プログラム名：ex05_2.cpp**）
List5-6 を修正し，double型で計算を行い，最後に分散（＝（値ー平均値）の2乗の平均値）も表示させるプログラムにする.

【実行例】

例1

入力：

```
1番の点数：1
2番の点数：2
3番の点数：3
4番の点数：4
5番の点数：5
```

出力：→

```
分散 = 2
```

例2

入力：

```
1番の点数：0
2番の点数：0
3番の点数：0
4番の点数：1
5番の点数：1
```

出力：→

```
分散 = 0.24
```

課題

課題5-3 （**プログラム名：ex05_3.cpp**）
配列 a は昇順に並んでいます。
　int a[] = {1, 3, 9, 11, 25, 36};
キーボードから整数を1つ読み取り、配列 a とその整数を統合して昇順を保ったまま出力しなさい。

ヒント
1. 配列 a を先頭から走査し、各要素と入力値 x を比較して小さい方を先に出力する（x は一度だけ出力するようフラグで管理）。
2. 端のケース(x <= a[0], x >= a[n-1])に注意

【実行例】

例1
入力：
10⏎
出力：
1 3 9 10 11 25 36

例2
入力：
-1⏎
出力：
-1 1 3 9 11 25 36

例3
入力：
40⏎
出力：
1 3 9 11 25 36 40

⏎ : enter

課題

課題5-4 （**プログラム名：ex05_4.cpp**）
学校に100個のロッカー、100人の学生がいます。初めはすべて閉。
1人目はすべて開ける。
2人目は2,4,6,…,100番を閉にする。
3人目は3,6,9,…,99番のロッカーを現在の状態を反転（開⇄閉）。
4人目は4,8,12,…,100番のロッカーを現在の状態を反転（開⇄閉）。
…同様に、100人目は100番を反転。
全員が終わったあと、開いているロッカーの番号を番号昇順・半角スペース区切りで出力しなさい。

ヒント：各ロッカーが反転された回数を配列で数え、奇数回なら開。

【実行例】

出力：