錬成問題

▪ 配列は、同一型の要素が直線上に並んだデータ構造である。配列内の任意の要素を参照するための演算子 **[]** の名称は [(1)] 演算子である。この演算子を適用した式 *a*[*i*] は、配列 *a* における先頭から [(2)] 個後ろの要素をアクセスする。要素の位置を指定するための **[]** 内の整数値のことを [(3)] と呼ぶ。

▪ 右のように宣言された配列を考える。*a* の型は [(4)] で、*a*[0] の型は [(5)] である。また、*b* の型は [(6)] で、*b*[0] の型は [(7)] で、*b*[0][0] の型は [(8)] である。なお、構成要素 *b*[2][2] の一つ前（先頭側）の構成要素は [(9)] であり、一つ後ろ（末尾側）の構成要素は [(10)] である。

```
int a[5];
int b[4][3];
```

▪ 配列内の要素を一つずつ順になぞっていく手続きのことを [(11)] と呼ぶ。

▪ 1 次元配列 *a* の要素数は [(12)] / [(13)] の除算によって求められる。また、2 次元配列 *b* の行数は [(14)] によって求められ、列数は [(15)] によって求められる。

▪ 以下に示すのは、要素の値が先頭から順に 1，2，3 である配列の宣言である。

```
int a[3] = (16);
```

▪ 以下に示すのは、全構成要素の値が 0 である 3 行 4 列の 2 次元配列の宣言である。

```
int b[(17)][(18)] = (19);
```

▪ 右に示すのは、要素数 5 の **int** 型配列 *a* の要素に、先頭から順に 10，20，30，40，50 を代入するプログラムである。

```
int a[(20)];
for (int i = 0; i < 5; i++)
    x[i] = (21);
```

▪ 右に示すのは、要素数 *n* の **float** 型配列 *a* の全要素の合計を変数 *sum* に格納するプログラムである。

```
float sum = (22);
for (int i = 0; i < (23); i++)
    sum += (24);
```

▪ 以下に示すのは、要素数が *n* である **int** 型配列 *a* の要素の最大値と最小値の差を求めて表示するプログラムである。

```
int min = (25);
int max = (26);
for (int i = (27); i < n; i++) {
    if ( (28) < min) min = (28);
    if ( (29) > max) max = (29);
}
cout << "最大値と最小値の差は" << (30) << "です。\n";
```

▪ 以下に示すのは、**double** 型配列 *a* の全要素の並びを反転するプログラムである（要素の値が先頭から順に 1.0，2.0，3.0 であれば 3.0，2.0，1.0 にする）。

```
for (int i = 0; i < (31) / 2; i++) {
    double t = a[i];
    a[i] = a[ (32) ];
    a[ (32) ] = (33);
}
```

▪ 以下に示すのは、**int** 型配列 *a* の全要素の並びをシャッフルする（要素の並びがランダムになるようにかき混ぜる）プログラムである。

```
for (int i = 0; i < (34); i++) {
    int j = (35);
    if (i != j) {
        int t = a[i];  a[i] = a[ (36) ];    a[ (36) ] = (37);
    }
}
```

▪ 以下に示すのは、**int** 型配列 *a* の全要素の値を先頭から順にコンマで区切って表示するプログラムである。たとえば、要素の値が先頭から順に 1，2，3 であれば「1，2，3」と表示する（3 の後にスペースやコンマを出力しない）。

```
int n = (38);
if (n >= (39))
    for (int i = 0; i < (40); i++)
        cout << a[ (41) ] << ", ";
cout << a[ (42) ];
```

▪ 以下に示すのは、２次元配列 *a* の全要素の値を表示するプログラムである（実行例は、２行１列と３行５列の出力例）。

```
cout << "{\n";
for (int i = 0; i < (43); i++) {
    cout << "   {";
    int n = (44);
    if (n >= 2)
        for (int j = 0; j < (45); j++)
            cout << (46) <<  ", ";
    cout << (47) << "},\n";
}
cout << "}\n";
```

```
{
   {1},
   {5},
}
```

```
{
   {1, 2, 3, 4, 5},
   {5, 3, 9, 7, 9},
   {0, 4, 6, 1, 8},
}
```

▪ 以下に示すのは、**int** 型配列 *a* の全要素の値を、記号文字 * を横に並べたグラフで表示するプログラムである（実行例に示すのは要素が {3，5，2，7，6} である場合）。

```
for (int i = 0; i < (48); i++) {
    cout << "a[" << (49) << "] : ";
    for (int j = 0; j < (50); j++)
        cout << '*';
    cout << '\n';
}
```

```
a[0] : ***
a[1] : *****
a[2] : **
a[3] : *******
a[4] : ******
```

▪ 以下に示す各プログラム部分の実行結果を示せ。

```
const int n = 5;
int a[n] = {0};
for (int i = 0; i < n; i++)
    cout << a[i] << ' ';
```

(51)

```
int b[3][2] = {{0,},{1,},};
for (int i = 0; i < sizeof(b) / sizeof(b[0]); i++)
    for (int j = 0; j < sizeof(b[0]) / sizeof(b[0][0]); j++)
        cout << b[i][j] << ' ';
```

(52)

▪ 右に示すのは、要素数 5 の **int** 型配列 *a* の全要素を配列 *b* にコピーするプログラムである。

```
for (int i = 0; i < 5; i++)
    b[i] = (53);
```

▪ 右に示すのは、要素数 *n* の **int** 型配列 *a* の要素のうち正の要素のみを配列 *b* に順にコピーするプログラムである。なお、変数 *c* は、コピーした要素数（正の要素数）である。

```
int c = 0;
for (int i = 0; i < n; i++) {
    if (a[i] > 0)
        b[(54)] = (55);
}
```

▪ 以下に示すのは、**int** 型配列 *a* の全要素の値（値はすべて正であるとする）を、記号文字 * を縦に並べた下向きグラフで表示するプログラムである（実行例に示すのは要素が {3, 5, 2, 7, 8, 4, 1, 9, 1, 0, 3, 4, 5} である場合）。なお、最初の行に出力するのは、添字の最下位桁である。

```
int max = (56);
for (int i = 1; i < sizeof(a) / sizeof(a[0]); i++)
    if (a[i] > (57)) (58) = a[i];

for (int i = 0; i < sizeof(a) / sizeof(a[0]); i++)
    cout << (59);
cout << '\n';

for (int i = 1; i <= (60); i++) {
    for (int j = 0; j < (61); j++)
        cout << (a[j] >= (62) ? '*' : ' ');
    cout << '\n';
}
```

```
0123456789012
********* ***
****** *  ***
** *** *  ***
 * *** *   **
 * **  *    *
   **  *
   **  *
    *  *
       *
```

▪ 以下に示すのは、要素数 10 の **int** 型配列 *a* の要素に 0 ～ 9 の 10 個の値をランダムな順序で格納する（たとえば {2, 0, 3, 5, 4, 8, 7, 9, 6, 1} とする）プログラムである。

```
for (int i = 0; i < (63); i++)
    a[i] = i;

for (int i = 9; i > (64); i--) {
    int j = rand() % ((65));
    if (i != j) {
        int t = a[i];
        a[(66)] = a[j];
        a[(67)] = t;
    }
}
```