



プログラミング言語 II

Programming Language II

— 2001年度版 —

福岡工業大学
情報工学部 情報工学科

柴田望洋

BohYoh Shibata
Fukuoka Institute of Technology

本資料について

- ◆ 本資料は、2001年度・福岡工業大学 情報工学部 情報工学科2年生の講義

『プログラミング言語』

の補助テキストとして、福岡工業大学 情報工学部 情報工学科 柴田望洋が編んだものである。

- ◆ 参考文献・引用文献等は、資料の最後にまとめて示す。
- ◆ 諸君が本資料をファイルに綴じやすいように、研究室の学生達（卒研究生と大学院生）が時間を割いて、わざわざ穴を開けるという作業を行っている（一度のパンチで開けることのできる枚数は限られており、気の遠くなるような時間がかかっている）。
必ずB5のバインダーを用意して、きちんと綴じていただきたい。
- ◆ 本資料のプログラムを含むすべての内容は、著作権法上の保護を受けており、著作権者である柴田望洋の許諾を得ることなく、無断で複写・複製をすることは禁じられている。

Microsoft

Microsoft Word 2000

まずは復習です！

課題 数当てゲームを作成する。

■ 第1版 変数に格納された値を当てさせる

```
/*
 数当てゲーム (その1)
*/

#include <iostream.h>

int main(void)
{
    int x;                // 読み込んだ値
    int no = 7;          // この数を当てさせる

    cout << "整数を入力せよ：";
    cin >> x;

    if (x > no)
        cout << "%a大きいです。 %n";
    else if (x < no)
        cout << "%a小さいです。 %n";
    else
        cout << "正解です。 %n";

    return (0);
}
```

if文の利用例です。読み込んだ値 x が、当てべき値 no より大きいか、小さいか、等しいかの判断を行います。

$\%a$ は、警報でしたね。通常は、ビープ音がなります。

※※※ このプログラムは、数を入力し、その判断を行うのが、1回限りです。当たるまで繰り返そうとする場合は、毎回プログラムを起動しなければなりません。

■ 第2版 変数に格納された値を当てさせる (当たるまで繰り返す)

```
/*
 数当てゲーム (その2)
*/

#include <iostream.h>

int main(void)
{
    int x;                // 読み込んだ値
    int no = 7;          // この数を当てさせる

    do {
        cout << "整数を入力せよ:";
        cin >> x;

        if (x > no)
            cout << "%a大きいです。%n";
        else if (x < no)
            cout << "%a小さいです。%n";
    } while (x != no);    // 当たっていない間繰り返す

    cout << "正解です。%n";

    return (0);
}
```

do 文を利用して、繰り返しを行います。繰り返しを続けるかどうかの判定は($x \neq no$)です。すなわち、読み込んだ値 x が、当てるべき値 no と等しくない間、繰り返します。

do 文による繰り返しが終了したときは、 x と no は等しくなっていますので、『正解です。』と表示します。

※※※ ここまでのプログラムは、当てさせるべき数がプログラム中に埋め込まれています。もし、7以外の数を当てさせようと思ったら、プログラムを書きかえて、コンパイル・実行し直さなければなりません。

■ 第3版 当てさせる数を乱数にする

```
/*
   数当てゲーム (その3)
*/
#include <time.h>
#include <stdlib.h>
#include <iostream.h>

int main(void)
{
    srand(time(NULL));           // 乱数の種を初期化
    int x;                       // 読み込んだ値
    int no = rand() % 1000;      // 0~999の乱数を発生

    do {
        cout << "整数を入力せよ:";
        cin >> x;

        if (x > no)
            cout << "%a大きいです。%n";
        else if (x < no)
            cout << "%a小さいです。%n";
    } while (x != no);          // 当たっていない間繰り返す

    cout << "正解です。%n";

    return (0);
}
```

srand 関数および rand 関数は、乱数発生のための関数です。

- srand 関数の呼出しを省略すると、プログラムを起動するたびに、同じ乱数が発生されてしまいます。実際に確認してみましょう。
- 発生した乱数を 1000 で割ったあまりが no に代入されます。したがって、0~999 を当てさせることとなります。この値を変更してみましょう。

★★★ このような関数の仕様については、

柴田望洋後援会オフィシャルホームページ <http://www.BohYoh.com/>
で学習することができます。

※※※ 何度も入力していれば、当たるのは当然です。入力できる回数に制限を付けると面白いですね。

■ 第4版 入力回数に制限を付ける

```
/*
 数当てゲーム (その4)
*/

#include <time.h>
#include <stdlib.h>
#include <iostream.h>

int main(void)
{
    srand(time(NULL));          // 乱数の種を初期化

    const int max = 10;        // 入力制限回数
    int x;                      // 読み込んだ値
    int no = rand() % 1000;     // 0~999の乱数を発生
    int cnt = max;              // 残り何回入力できるか?

    do {
        cout << "残り" << cnt << "回です。¥n";
        cout << "整数を入力せよ：";
        cin >> x;
        cnt--;

        if (x > no)
            cout << "¥a大きいです。¥n";
        else if (x < no)
            cout << "¥a小さいです。¥n";
    } while (x != no && cnt > 0);

    if (x == no) {
        cout << "正解です。¥n";
        cout << max - cnt << "回で当たりましたね。¥n";
    }

    return (0);
}
```

`max` は何回挑戦できるかを表す定数です。`cnt` は、残り何回挑戦できるかを表す変数です。最初は、`cnt` の値は `max` と同じ値です (この場合 10 回)。入力するたびに、`cnt` の値をデクリメントしていきます。

`do` 文の判定に、`cnt > 0` が追加されていることに注意しましょう。残り挑戦回数を使い果たすと、`do` 文は終了です。

★課題★

いったん数当てが終了したら (成功でも / 失敗でも)、再び繰り返すかどうかを尋ね、繰り返しせよるようにせよ。終了時は、全体で何回成功したかを表示すること。

復習の続きです！

課題 for 文と文字による描画を行う

■ *を横に連続表示

```
/*
   for文+文字による描画 *を横に連続表示
*/
#include <iostream.h>

int main(void)
{
    int i;
    int no;           // 表示する個数

    cout << "いくつ表示しますか：";
    cin >> no;

    for (i = 0; i < no; i++)
        cout << '*';
    cout << '\n';

    return (0);
}
```

```
いくつ表示しますか：15
*****
```

■ +-を横に連続表示

```
/*
   for文+文字による描画 +-を横に連続表示
*/
#include <iostream.h>

int main(void)
{
    int i;
    int no;           // 表示する個数

    cout << "いくつ表示しますか：";
    cin >> no;

    for (i = 0; i < no; i++)
        if (i % 2)
            cout << '-';
        else
```

```
いくつ表示しますか：15
+-+-+---+---+---
```

```
        cout << '+';\n    cout << '\\n';\n\n    return (0);\n}
```

for 文による繰返しにおいて、偶数と奇数のときとで、+を表示するか、-を表示するかを切り替えます。

■ 0123456789 を横に連続表示

```
/*\n for文+文字による描画 0123456789を横に連続表示\n*/\n#include <iostream.h>\n\nint main(void)\n{\n    int i;\n    int no;                // 表示する個数\n\n    cout << "いくつ表示しますか：";\n    cin >> no;\n\n    for (i = 0; i < no; i++)\n        cout << i % 10;\n    cout << '\\n';\n\n    return (0);\n}
```

```
いくつ表示しますか：15\n012345678901234
```

for 文による繰返しにおいて、10 で割った剰余を表示します。

★課題★

1234567890 と表示するように変更せよ。たとえば、15 であれば、

```
123456789012345
```

と表示する。

課題 for 文と文字による描画を行う (長方形)

■ *で長方形を表示

<pre> /* for文+文字による描画 *で長方形を表示 */ #include <iostream.h> int main(void) { int i, j; int height, width; // 行数・列数 cout << "行数は："; cin >> height; cout << "列数は："; cin >> width; for (i = 0; i < height; i++) { for (j = 0; j < width; j++) cout << '*'; cout << '\n'; } return (0); } </pre>	<pre> 行数は：12 列数は：21 ***** ***** ***** ***** ***** ***** ***** ***** ***** ***** ***** ***** </pre>
--	--

■ +-で長方形を表示

<pre> /* for文+文字による描画 +-で長方形を表示 */ #include <iostream.h> int main(void) { int i, j; int height, width; // 行数・列数 cout << "行数は："; cin >> height; cout << "列数は："; cin >> width; for (i = 0; i < height; i++) { for (j = 0; j < width; j++) if ((i + j) % 2) cout << '-'; else </pre>	<pre> 行数は：7 列数は：21 +-+-+-+ +-+-+-+ +-+-+-+ +-+-+-+ +-+-+-+ +-+-+-+ +-+-+-+ </pre>
--	---

```
        cout << '+';
        cout << '\n';
    }

    return (0);
}
```

■ 0123456789 で長方形を表示

```
/*
   for文+文字による描画 0123456789で長方形を表示
*/
#include <iostream.h>

int main(void)
{
    int i, j;
    int height, width;           // 行数・列数

    cout << "行数は : ";      cin >> height;
    cout << "列数は : ";      cin >> width;

    for (i = 0; i < height; i++) {
        for (j = 0; j < width; j++)
            cout << (i + j) % 10;
        cout << '\n';
    }

    return (0);
}
```

```
行数は : 12
列数は : 21
0123456789012345678901
1234567890123456789012
2345678901234567890123
3456789012345678901234
4567890123456789012345
5678901234567890123456
6789012345678901234567
7890123456789012345678
8901234567890123456789
9012345678901234567890
0123456789012345678901
1234567890123456789012
```

★課題★

数字の平行四辺形を表示するように変更せよ。たとえば、行数が7で列数が21であれば、

```
0123456789012345678901
 2345678901234567890123
   4567890123456789012345
    6789012345678901234567
     8901234567890123456789
      0123456789012345678901
       2345678901234567890123
```

と表示する。

課題 配列を活用する

■ 読み込んだ値を表示

```
/*
  配列：読み込んだ値を表示
*/

#include <iostream.h>

int main(void)
{
    const int num = 10;          // 要素数
    int i;
    int a[num];                 // 配列

    cout << "各要素の値を入力せよ。¥n";
    for (i = 0; i < num; i++) {
        cout << "a[" << i << "] : ";
        cin >> a[i];
    }

    for (i = 0; i < num; i++)
        cout << "a[" << i << "] = " << a[i] << "¥n";

    return (0);
}
```

各要素の値を入力せよ。

```
a[0] : 12
a[1] : 75
a[2] : 6
a[3] : 23
a[4] : 12
a[5] : 128
a[6] : 256
a[7] : 9
a[8] : 12
a[9] : 33
a[0] = 12
a[1] = 75
a[2] = 6
a[3] = 23
a[4] = 12
a[5] = 128
a[6] = 256
a[7] = 9
a[8] = 12
a[9] = 33
```

■ 最小値を表示

```
/*
  配列：最小値を表示
*/

#include <iostream.h>

int main(void)
{
    const int num = 10;          // 要素数
    int i;
    int a[num];                 // 配列
    int min;                     // 最小値

    cout << "各要素の値を入力せよ。¥n";
    for (i = 0; i < num; i++) {
        cout << "a[" << i << "] : ";
        cin >> a[i];
    }
}
```

各要素の値を入力せよ。

```
a[0] : 12
a[1] : 75
a[2] : 6
a[3] : 23
a[4] : 12
a[5] : 128
a[6] : 256
a[7] : 9
a[8] : 12
a[9] : 33
最小値 : 6
```

```

    min = a[0];
    for (i = 1; i < num; i++)
        if (a[i] < min) min = a[i];

    cout << "最小値：" << min << '\n';

    return (0);
}

```

■ ある値をもつ要素をすべて表示

```

/*
 配列：ある値をもつ要素をすべて表示
*/
#include <iostream.h>

int main(void)
{
    const int num = 10;          // 要素数
    int i, x, cnt;
    int a[num];                  // 配列

    cout << "各要素の値を入力せよ。¥n";
    for (i = 0; i < num; i++) {
        cout << "a[" << i << "] : ";
        cin >> a[i];
    }
    cout << "調べる値は：";
    cin >> x;

    cnt = 0;
    for (i = 0; i < num; i++) {
        if (a[i] == x) {
            cnt++;
            cout << "a[" << i << "] = " << a[i] << '\n';
        }
    }
    cout << "該当する要素の個数=" << cnt << '\n';

    return (0);
}

```

各要素の値を入力せよ。

a[0] : 12

a[1] : 75

a[2] : 6

a[3] : 23

a[4] : 12

a[5] : 128

a[6] : 256

a[7] : 9

a[8] : 12

a[9] : 33

調べる値は：12

a[0] = 12

a[4] = 12

a[8] = 12

該当する要素の個数=3

★課題★

逆順に表示せよ。

課題 配列を活用する (続き)

50 個の要素をもつ配列に、0~9 の乱数を格納し、その分布を表示する。

■ 第 1 版 : 数で表示

```
/*
   50個の0~9の乱数を生成して分布を表示
*/
#include <time.h>
#include <stdlib.h>
#include <iostream.h>

int main(void)
{
    int i;
    const int max = 50;      // 乱数を発生させる個数
    int x[max];             // 配列
    int b[10] = {0};        // 分布

    srand(time(NULL));      // 乱数の種を初期化

    for (i = 0; i < max; i++) {
        int r = rand() % 10; // 0~9の乱数を発生
        x[i] = r;
        b[r]++;
    }

    for (i = 0; i < 10; i++)
        cout << i << "は" << b[i] << "個\n";

    return (0);
}
```

0は3個
1は8個
2は6個
3は3個
4は9個
5は2個
6は5個
7は5個
8は3個
9は6個

配列 **x** には、50 個の乱数が格納されます。配列 **b** の各要素には分布が格納されます。たとえば、**b[3]** の値は、配列 **x** の中で値が 3 である要素数です。

※乱数を発生させますから、実行によって得られる分布は、この通りにはなりません。

■■■ BohYoh.com ■■■

`rand` 関数や `srand` 関数の仕様、これらの用法などについては、柴田望洋後援会オフィシャルホームページ

<http://www.BohYoh.com/>

の『C言語講座』『C/C++ FAQ』などのページを参照してください。

*を並べて表示するプログラムを学習しましたから、その技術を使って、グラフとして分布を表示してみましょう。

■ 第2版：グラフで表示

```
/*
   50個の0～9の乱数を生成して分布を表示（横向き棒グラフ）
*/
#include <time.h>
#include <stdlib.h>
#include <iostream.h>

int main(void)
{
    int i, j;
    const int max = 50;          // 乱数を発生させる個数
    int x[max];                 // 配列
    int b[10] = {0};           // 分布

    srand(time(NULL));         // 乱数の種を初期化

    for (i = 0; i < max; i++) {
        int r = rand() % 10;    // 0～9の乱数を発生
        x[i] = r;
        b[r]++;
    }

    for (i = 0; i < 10; i++) {
        cout << i << " : ";
        for (j = 0; j < b[i]; j++)
            cout << '*';
        cout << '\n';
    }

    return (0);
}
```

```
0 : ***
1 : ***
2 : *
3 : *****
4 : *****
5 : *****
6 : *****
7 : *****
8 : *****
9 : ***
```

次は、縦向きグラフで表示します。少し難しくなります。

■ 第2版：グラフで表示

```

/*
 50個の0~9の乱数を生成して分布を表示（縦向き棒グラフ）
*/

#include <time.h>
#include <stdlib.h>
#include <iostream.h>

int main(void)
{
    int i;
    const int max = 50;           // 乱数を発生させる個数
    int x[max];                  // 配列
    int b[10] = {0};             // 分布
    int bmax;                     // 分布の最大値

    srand(time(NULL));           // 乱数の種を初期化

    for (i = 0; i < max; i++) {
        int r = rand() % 10;     // 0~9の乱数を発生
        x[i] = r;
        b[r]++;
    }

    bmax = b[0];
    for (i = 1; i < 10; i++)
        if (b[i] > bmax) bmax = b[i];

    while (bmax > 0) {
        for (i = 0; i < 10; i++)
            if (b[i] >= bmax)
                cout << " * ";
            else
                cout << "   ";
        cout << '\n';
        bmax--;
    }

    cout << "-----\n";
    cout << " 0  1  2  3  4  5  6  7  8  9 \n";

    return (0);
}

```

