計算機基礎実習 I ・第 13 回

2017 年 7 月 14 日

今回の内容

13.1 ゲームを作ろう

「計算機基礎実習 I」も残すところ 3 回となりました。今回は、これまでの授業で勉強したことを応用して、次のようなゲームを作ってみることにします。

数当てゲーム このゲームは、コンピュータ (このゲームプログラム) が勝手に選んだ整数を、ユーザーが推測して当てるというものです。

1. まず、コンピュータは 1 ～ 9 の 9 種類の数字から勝手に 3 種類の数字を選び、この 3 つを勝手に並べて 3 桁の整数を作ります。つまり、この 3 桁の整数には 0 という数字は使われていませんし、同じ数字が 2 度以上使われていることもありません。コンピュータの選んだ 3 桁の整数が問題の正解になります。
2. ユーザーは、コンピュータの選んだ 3 桁の整数を推測し、それを入力します。最初は全く手がかりがありませんので適当に整数を選んで入力しなければなりません。入力として許されるのは、コンピュータが選ぶことのできる形の 3 桁の整数でなければなりません。つまり、0 を含んでいたり、同じ数字を 2 度以上使っているような整数を入力してはいけません。
3. ユーザーが 3 桁の整数を入力すると、コンピュータは次のような 2 つのヒントを返します。

   Hit 数　　数字が正解と一致しているような桁の個数
   Blow 数　正解では別の桁にその数字が使われているような桁の個数

   たとえば、正解が 123 で、ユーザーの推測 (入力) が 132 の場合は、Hit 数は 1、Blow 数は 2 となります。
4. 正解が入力される (Hit 数が 3 になる) まで、2 と 3 が繰り返されます。ユーザーは 10 回以内に正解を入力しなければなりません。

```
1回目 ? 123
1H0B
2回目 ? 456
0H0B
3回目 ? 789
0H2B
4回目 ? 178
0H1B
5回目 ? 923
1H1B
6回目 ? 827
1H0B
7回目 ? 893
おめでとう。正解です。
```

右は、このゲームプログラム (`hitblow.c`) の実行例です。コンピュータが選んだ数 (正解) は 893 でした。

「··· 回目 ?」というプロンプト (入力を促すメッセージ) の後の 3 桁の整数がユーザーの推測 (入力) です。入力された整数の Hit 数が n、Blow 数が m の場合、プログラムは nHmB のようにヒントを表示しています。7 回目の入力 893 は正解でしたので、ここでプログラムが終了しています。

メモ

数当てゲームのプログラム　では、このゲームプログラムのソースファイル hitblow.c を紹介します。ただし、行頭の整数は説明のために付けた行番号で、このプログラムの一部ではありません。また、ソースプログラム中で /* と */ とで囲まれた部分は注釈 (プログラムの説明) です。C 言語では、このようにしてプログラム中に任意の注釈を書いておくことができます。この部分はプログラムとしては無視されてコンパイルされます。

hitblow.c

```
 1 #include <turtle.h>
 2 #include <stdlib.h>
 3 #include <time.h>
 4
 5 main()
 6 {
 7     int try, hit, blow;
 8     int x1, x2, x3, g1, g2, g3;
 9     int guess;
10
11     /* 現在時刻を乱数の種として設定 */
12     srand(time(0));
13
14     /* 正解(1〜9の数字3個の重複のない並び)を乱数で生成する */
15     x1 = rand()%9+1;                    /* 左端 */
16     do {
17         x2 = rand()%9+1;                /* 真中 */
18     } while (x2 == x1);                 /* 同じ数だったらもう一度 */
19     do {
20         x3 = rand()%9+1;                /* 右端 */
21     } while (x3 == x1 || x3 == x2);     /* 同じ数だったらもう一度 */
22
23     tPenUp();
24     tMoveTo(-100, 200);
25
26     for (try = 1; try <= 10; try++) {
27         /* キーボードから3桁の数を入力 */
28         tPrintf("%d回目 ? ", try);
29         tScanf("%d", &guess);
30
31         /* 各桁を取り出す */
32         g1 = guess/100;          /* 左端 */
33         g2 = guess/10%10;        /* 真中 */
34         g3 = guess%10;           /* 右端 */
35
36         /* 1〜9の数字3個の並びかどうかをチェックする */
37         if (g1 > 9 || g1 < 1 || g2 < 1 || g3 < 1)
38             continue;
39
40         /* その3個の数字に重複がないかどうかをチェックする */
41         if (g1 == g2 || g2 == g3 || g3 == g1)
```

```
42              continue;
43
44          /* hit と blow の数を数える */
45          hit = blow = 0;
46          if (g1 == x1) hit++;
47          if (g2 == x2) hit++;
48          if (g3 == x3) hit++;
49          if (g1 == x2 || g1 == x3) blow++;
50          if (g2 == x3 || g2 == x1) blow++;
51          if (g3 == x1 || g3 == x2) blow++;
52
53          /* 正解なら繰り返しを終了 */
54          if (hit == 3)
55              break;
56
57          /* hit と blow の数を表示 */
58          tPrintf("%dH%dB\n", hit, blow);
59      }
60
61      /* 10回以内で正解が入力されなかったら、正解を表示する。 */
62      if (try <= 10)
63          tPrintf("おめでとう。正解です。\n");
64      else
65          tPrintf("残念でした。正解は %d%d%d です。\n", x1, x2, x3);
66  }
```

メモ

乱数の発生　まず、正解となる 3 桁の整数を作らなければなりません。このプログラムでは、その整数の各桁を左から順に x1、x2、x3 という 3 つの変数に記憶しています。各変数には 1 から 9 までの数字を適当に選んで記憶しておかなければなりません。そこで必要となるのが**乱数**と呼ばれるでたらめな数です。C というプログラミング言語では rand() という式を使うことで、このでたらめな (0 以上の) 整数を生成することができます。たとえば、

```
x1 = rand();
```

とすれば、変数 x1 にでたらめな 0 以上の整数が格納されます。このゲームで必要なのは、1 から 9 までの数 (字) ですから、

```
x1 = rand()%9+1;
```

とすることでこれを実現できます。でたらめな整数を 9 で割ったあまりが 0 から 8 までの (でたらめな) 数となり、これに 1 を足すことで 1 から 9 までの (でたらめな) 数となるからです。これを行っているのが、hitblow.c の 15 行目です。

rand() という式の値は、それが計算される度にでたらめな整数となりますので、真中の桁の数字 x2 も同じように rand() を使って作ることができます。ただし、このゲームでは 3 桁の中に同

じ数字を 2 回以上使うことは許されませんから、x2 が x1 と同じになったら、もう一度

```
x2 = rand()%9+1;
```

を実行して、別の数字が生成されるまでこれを繰り返します。この繰り返しが 16 〜 18 行目の do 文です。右端の桁 x3 の場合は、同様に x1 とも x2 とも異なる数字が生成されるまで繰り返すことになります (19 〜 21 行目)。

メモ

乱数の種　rand() という式が生成する整数は、見かけは乱数のように見えますが、実はある規則にしたがって順に決った整数が生成されているに過ぎません。つまり、プログラムの実行が開始されてから rand() が 1 番目、2 番目、3 番目、... と生成していく乱数は常に同じ数列になります。これでは何度このゲームプログラムを実行しても、正解の 3 桁の整数は常に同じものになってしまいますので、hitblow.c の 12 行目では、現在時刻[1]を表す time(0) という式の値を**乱数の種**として設定するという処理を行っています。rand() が生成する乱数列は、srand によって設定された整数 (乱数の種) によって変りますので、プログラムが実行されたときの時刻に応じて正解も変ってくれるようになります。

プログラムの中で、乱数を発生する rand() や、乱数の種を設定する srand() を使う場合は、プログラムの冒頭部分に「#include <stdlib.h>」を、現在時刻を取得する time() を使う場合は「#include <time.h>」を書いておく決まりになっています (2 〜 3 行目)。

メモ

13.2　continue 文

hitblow.c の 36 行目から 42 行目では、キーボードから入力された数が、1 から 9 までの数字 3 個の重複のない並びであるかどうかをチェックしています。このチェックを通らなかった場合に実行されている continue; という文は、そのとき while や for、do によって繰り返されている文 (ブロック) の残りの部分を飛び越して、その回の実行をそこで終えるための構文です。

for 文の中で continue; が実行された場合、第 12 回で説明した「再設定式」(hitblow.c の場合は 26 行目の try++) が実行されてから、「継続条件式」が調べられます。もし、「継続条件式」が成り立っていれば、繰り返しの次の回が開始されます。成り立っていなければ、for 文の実行は終

[1] 世界標準時の 1970 年 1 月 1 日 00:00:00 からの経過秒数で現在時刻を表します。

了します。また、while 文や do 文の中で continue 文が実行された場合は、すぐに、その while 文や do 文の条件式が調べられ、成り立っていれば、繰り返しの次の回が開始されます。成り立っていなければ、while 文や do 文の実行は終了します。

メモ

13.3 演習問題

1. hitblow.c を hitblow0.c にコピーし、この hitblow0.c を、正解 3 桁の中に 0 を含んでもよいようなルールに改造し、コンパイル、実行して正しく動作することを確認しなさい。ただし、「32」のような 2 桁の整数も、100 の位が 0 である 3 桁とみなして、正解となりうるものとします。正解が「32」の場合、ユーザーが「32」と入力しても、「032」と入力しても、正解とみなしてください。

```
1回目 ? 123
0H2B
2回目 ? 456
0H0B
3回目 ? 789
0H0B
4回目 ? 032
0H2B
5回目 ? 031
1H1B
6回目 ? 012
0H3B
7回目 ? 201
おめでとう。正解です。
```

ヒント ユーザーが、たとえば「032」をキーボードから入力した場合でも、

```
tScanf("%d", &guess);
```

は、guess に整数値 32 を代入してくれます。

13.4 今回の実習内容

1. プリントをもう一度読み返しましょう。例題 hitblow.c のソースプログラムを作成し、コンパイル、実行してみましょう。プログラムが完成したら「課題の提出と確認」の Web ページから提出してください。
2. 演習問題に取り組みましょう。プログラムが完成したら、「課題の提出と確認」の Web ページからの提出を忘れないでください。
3. クイズに答えてください。前回までと同様に「課題の提出と確認」の Web ページで「第 13 回クイズ」を選択し、「送信」のボタンをクリックしてクイズに答えてください。

4. 以下の要領で、自由に魅力的なプログラムを作成してください。

- ソースファイルの名前は `final.c` としてください。
- もちろん、マウスやキーボードからの入力を使っても ok です。ただし、どのように操作すればよいか、そのプログラムを実行しただけで分かるように工夫してください。
- ソースファイルには他人が書いたプログラムが含まれていてはいけません。
- この科目で紹介したことのない関数は (基本的には) 使用できません。もし、どうしても使用したい場合は早めに担当教員に相談してください。
- プログラムが完成したら、第 15 回 (7 月 28 日) の授業開始以降に「課題の提出と確認」の Web ページから提出してください。
- 提出の締め切りは 7 月 31 日 (月) 18:30 です。
- みなさんが作ったプログラム (ソースプログラムとオブジェクトプログラム) は、8 月 2 日 (水) 12:00 から 8 月 4 日 (金) 18:30 の間、匿名で公開 (学内のみ) します。
- 公開期間中、1 人 10 件程度、他の受講者が作成したプログラムを評価して頂きます。この評価も課題の一部です。セルフラーニング室等を利用して、「課題の提出と確認」の Web ページから、8 月 4 日 (金) 18:30 までに忘れず評価を行ってください。
- 評価は、次のような観点で行い、自分が評価を担当していないプログラムも含めて、ごく平均的と思われるプログラムの点数が、満点の半分になるように採点してください。
  (a) プログラムを実行した時の動作や、その描く図形などは魅力的か (10 点満点)
  (b) ソースプログラムにこの科目で勉強したことが反映されているか (5 点満点)

次のフォルダに昨年度の作品が置いてありますので、参考にしてください。

`R:\a89023\`計算機基礎実習I`\2016`年度自由課題

みなさんが作成した作品も、来年度以降の受講者に対して同様に紹介させていただきます。

計算機基礎実習 I ・第 13 回・終わり

付録：前回の演習問題のプログラム例

rpolygon.c

```
 1 #include <turtle.h>
 2
 3 main ()
 4 {
 5     int n, i;
 6
 7     tGetClick();
 8     n = tClickCount();
 9     for (i = 0; i < n; i++) {
10         tForward(100);
11         tArc(30, 360.0/n);
12     }
13 }
```

dots.c

```
 1 #include <turtle.h>
 2
 3 main ()
 4 {
 5     tSetColor(1.0, 0.0, 0.1);
 6     tPenUp();
 7     while (tGetClick() == 1) {
 8         tMoveTo(tClickX(), tClickY());
 9         tForward(20);
10         tTurn(-90);
11         tMark();
12         tCircle(-20);
13         tFill();
14     }
15     tMoveTo(0, 0);
16 }
```