第３回

５．ループ（繰り返し）

〔１〕for 文

例題１－１５

正の数 n をキーボードから入力すると，1 から n までの整数の和を出力するプログラムをつくりなさい。

▼出力結果１－１５

2 以上の整数を入力してください！
357↵
1 +・・・+ 357 = 63903

考え方

合計を保持する変数を long 型で宣言し，入力した数値 n までループ処理で累積する。正しい数値が入力されたとき（n は 2 以上）だけ結果を出力する。n が 2 未満のときは，なにも出力しないこととする。

▼プログラム１－１５

```
01      /* E1-15 */
02      /* １からｎまでの整数の和 */
03      #include <stdio.h>
04
05      main()
06      {
07          long i, s, n;
08          s = 0;
09          printf("2 以上の整数を入力してください！¥n");
10          scanf("%ld",&n);
11          if (n >= 2) {
12              for (i = 1; i <= n; ++i)
13                  s += i;
14              printf("1 +‥‥+ %ld = %ld¥n", n, s);
15          }
16      }
```

練習問題１１

１１-１．正の整数ｎをキーボードから入力すると，ｎの階乗を出力するプログラムをつくりなさい。ｎが正でない場合は入力のやり直しをさせるようにすること。

ｎの階乗 n! = n×(n - 1)×・・・×2×1　ただし，0! = 1

１１-１２．二つの正の整数ｍ, ｎをキーボードから入力すると，ｍからｎまでの和を出力するプログラムをつくりなさい。ただし，正整数ｍ，ｎが ｍ > ｎで入力された場合は，ｎからｍまでの和を出力するものとする。また，ｍ，ｎが正でない場合は入力のやり直しをさせるようにすること。

▼出力例

正の異なる整数を二つ入力してください！

12 35↵

12 + … + 35 = 564

〔２〕while 文

例題１－１６

キーボードから 1 文字を入力すると，これをそのまま出力するプログラムをつくりなさい。

▼出力結果１－１６

```
文字を入力してください！
abcde FGHIJ 123↵
abcde FGHIJ 123
Ctrl-d↵      <--  Linux での入力終了     cf. Windows では Ctrl-z↵
```

考え方

1 文字を入力して，これが EOF でなければそのまま 1 文字を出力する。同じ処理を繰り返し，入力された文字が EOF ならば終了する。stdio.h に EOF が定義されているので，改めて定義する必要はない。

▼プログラム１－１６

```
01      /* E1-16-1 */
02      /* １文字入出力 */
03      #include <stdio.h>
04
05      main()
06      {
07          char c;
08          printf("文字を入力してください！¥n");
09          c = getchar();
10          while (c != EOF) {
11              putchar(c);
12              c = getchar();
13          }
14      }
```

《別解》

```
01      /* E1-16-2 */
02      /* １文字入出力 */
03      #include <stdio.h>
04
05      main()
06      {
07          int c;
08          printf("文字を入力してください！¥n");
09          while ((c = getchar()) != EOF)
10              putchar(c);
11      }
```

練習問題１２

１２−１．正の整数 n をキーボードから入力すると，1 から n までの奇数の和を出力するプログラムをつくります。while 文を用いてつくりなさい。ただし，n が正であるかどうか判定して，正でなければ入力のやり直しを要求するようにしなさい。

１２−２．実数型の float と double の変数を１つずつ宣言しなさい。どちらの変数も 0 で初期化してから,それぞれ 0.01 を 1 万回加えてその結果を比較しなさい。float と double で計算結果に差が出るとしたら，その理由を述べなさい。

《ヒント》浮動小数点数演算での定数を定義してあるヘッダファイル<float.h>を調べると，float や double で表現できる最小数が分かります。

〔３〕do ～ while 文

次のループ処理で，文１，文２をそれぞれ 10 回ずつ実行するように，①，②に条件式を書きなさい。

```
i = 1;
while (①) {
    文 1;
    ++i;
}
```

```
i = 1;
do {
    文２;
    i++;
} while (②);
```

〔４〕ループの入れ子（ネスト）

例題１－１７

九九の表を出力するプログラムをつくりなさい。ただし，出力が見やすいように，見出しや罫線をつけること。

▼出力結果１－１７

九九の計算

```
+---+--+--+--+--+--+--+--+--+
|   | 1| 2| 3| 4| 5| 6| 7| 8| 9|
+---+--+--+--+--+--+--+--+--+--+
| 1 | 1| 2| 3| 4| 5| 6| 7| 8| 9|
| 2 | 2| 4| 6| 8|10|12|14|16|18|
| 3 | 3| 6| 9|12|15|18|21|24|27|
| 4 | 4| 8|12|16|20|24|28|32|36|
| 5 | 5|10|15|20|25|30|35|40|45|
| 6 | 6|12|18|24|30|36|42|48|54|
| 7 | 7|14|21|28|35|42|49|56|63|
| 8 | 8|16|24|32|40|48|56|64|72|
| 9 | 9|18|27|36|45|54|63|72|81|
+---+--+--+--+--+--+--+--+--+--+
```

▼プログラム１－１７

```
01      /* E1-17 */
02      /* 九九の計算 */
03      #include <stdio.h>
04
05      main()
06      {
07          int i, j;
08          printf("九九の計算¥n");
09          printf("+---+--+--+--+--+--+--+--+--+--+¥n");
10          printf("|   | 1| 2| 3| 4| 5| 6| 7| 8| 9|¥n");
11          printf("+---+--+--+--+--+--+--+--+--+--+¥n");
12          for (i = 1; i < 10; ++i) {
13              printf("|%2d |",i);
14              for (j = 1; j < 10; ++j) {
15                  printf("%2d|", i*j);
16              }
17              printf("¥n");
18          }
19          printf("+---+--+--+--+--+--+--+--+--+--+¥n");
20      }
```

練習問題１３

１３−１．1 を除き，その数以外では割り切れない数を素数という。2 から 32767 までのすべての素数とその個数を出力するプログラムをつくりなさい。なお，出力の形式は下図のように，横方向に 10 桁表示で 5 個ずつとし，最後に素数の個数を示すものとする。

《ヒント》ある数を素数候補として，2 から順番に割り算を行い，その数でしか割り切れなかったときに素数と判定する。できるだけ効率的な判定方法を考えてみなさい。

```
         2         3         5         7        11
        13        17        19        23        29
         .         .         .         .         .
         .         .         .         .         .
         .         .         .         .         .
     32687     32693     32707     32713     32717
     32719     32749
素数は   3512 個ありました。
```

１３−２．下図のように左向き二等辺三角形を，文字「＊」を用いて出力するプログラムをつくりなさい。

《ヒント》for 文または while 文を入れ子にして，putchar(' ')と putchar('*')を繰り返す。制御回数に注意する。

```
         *
        **
       ***
      ****
     *****
    ******
   *******
  ********
 *********
********** <-- 10 文字，全体で 19 行
 *********
  ********
   *******
    ******
     *****
      ****
       ***
        **
         *
```

１３−３. 2 以上の整数を読み込み, その素因数分解を出力するプログラムをつくりなさい。
例えば, 出力結果は次のようになる。

▼出力例

2 以上の整数を入力してください (exit = 0) : 120↵
120 = 2×2×2×3×5

2 以上の整数を入力してください (exit = 0) : 1234↵
1234 = 2×617

2 以上の整数を入力してください (exit = 0) :

６．プリプロセッサ（1）

例題１－１８

#define 命令を使って，3.14159 を PAI，scanf を YOMU，printf を KAKU と定義して，さらに球の表面積と体積を求める式も引数付きマクロで定義してみましょう。そして，出力結果１－１８のように半径を入力すると，球の表面積と体積を小数第１位まで求めるプログラムをつくりなさい。次に，半径を下表に示した値として，その表を完成しなさい。

▼出力結果１－１８

半径ｒは？5.5↵
球の表面積 = 380.1
球の体　積 = 696.9

半径	6.6	13.5	24.8	31.1	42.8
球の表面積		2290.2			23019.6
球の体積	1204.3		63891.5	125999.7	

▼プログラム１－１８

```
01      /* E1-18 */
02      /* プリプロセッサ(1) */
03      #include <stdio.h>
04      #define PAI 3.14159
05      #define YOMU scanf
06      #define KAKU printf
07      #define SURFACE(r) (4.0 * PAI * (r) * (r))
08      #define VOLUME(r)  (4.0 / 3.0 * PAI * (r) * (r) * (r))
09      main()
10      {
11          float r;
12          KAKU("半径ｒは？");
13          YOMU("%f",&r);
14          KAKU("球の表面積 = %.1f¥n", SURFACE(r));
15          KAKU("球の体　積 = %.1f¥n", VOLUME(r));
16      }
```

《マクロの置換について》

次の形で定義する。

　　#define　*name*　　置換テキスト

例えば,

　　#define　　forever　　for (;;)

また，引数付きマクロの定義も可能である。例えば，すべてのデータ型で使える

　　#define　MAX(A, B)　　((A) > (B) ? (A) : (B))

が定義できる。

しかし，落とし穴もある。次のような使い方はどうなるだろうか？

　　MAX(i++, j++)

次の定義はどうだろうか？

　　#define　　SQUARE(x)　　x * x

間違っているとしたら，正しい定義はどうなるだろう。

７．ライブラリ

〔１〕数学関数

例題１－１９

出力結果１－１９のように，角度 x を 0，30，60，90，…，360 度のように 30 度ずつ変化させたときの sin(x)，cos(x)，tan(x)の値を小数第 3 位まで求めるプログラムをつくりなさい。

▼出力結果１－１９

```
x          sin(x)      cos(x)      tan(x)
-------------------------------------------
0.0         0.000       1.000       0.000
30.0        0.500       0.866       0.577
60.0        0.866       0.500       1.732
90.0        1.000       0.000  753695.995
120.0       0.866      -0.500      -1.732
150.0       0.500      -0.866      -0.577
180.0       0.000      -1.000      -0.000
210.0      -0.500      -0.866       0.577
240.0      -0.866      -0.500       1.732
270.0      -1.000      -0.000  251231.998
300.0      -0.866       0.500      -1.732
330.0      -0.500       0.866      -0.577
360.0      -0.000       1.000      -0.000
```

考え方

三角関数を使うときは，その角度は，すべてラジアンで表さなければならない。

度数 x をラジアン r の単位に変換するには，x*3.14159/180.0 で行う。

▼プログラム１－１９

```
01      /* E1-19 */
02      /* 数学関数 */
03      #include <stdio.h>
04      #include <math.h>
05      #define RAD (3.14159/180.0)
06
07      main()
08      {
09          double x, r, s, c, t;
10          printf("    x     sin(x)     cos(x)     tan(x)¥n");
11          printf("-------------------------------------¥n");
12          for (x = 0.0; x <= 360.0; x = x + 30.0) {
13              r = x * RAD;
14              s = sin(r);
15              c = cos(r);
16              t = tan(r);
17              printf("%5.1f %10.3f %10.3f %10.3f¥n", x, s, c, t);
18          }
19      }
```

《数学関数 math.h を利用する場合のコンパイルの注意》

gcc でコンパイルする際，オプション-lm を必ずつける。

```
$ gcc -o E1-19 E1-19.c -lm↵
```

これにより，数学関数ライブラリ libm.a がリンクされる。

練習問題１４

１４-1. 1 から 20 までの 2 乗，3 乗，平方根，立方根を求めるプログラムをつくりなさい。
《ヒント》数学関数の引数と戻り値は double で宣言されている点に注意しなさい。

１４-２. 直角三角形の底辺と高さを入力すると，斜辺の長さを小数第 1 位まで出力するプログラムを，三平方の定理を用いてつくりなさい。

〔２〕文字列変換関数

例題１－２０

文字列 "123", "123.45", "123456789" をそれぞれ数値化して，その値を 3 倍するプログラムをつくりなさい。

▼出力結果１－２０

```
369
370.35000
370370367
```

▼プログラム１－２０

```
01      /* E1-20 */
02      /* 文字列変換関数 */
03      #include <stdio.h>
04      #include <stdlib.h>
05
06      main()
07      {
08          int a, x;
09          double b, y;
10          long c, z;
11          a = atoi("123");
12          b = atof("123.45");
13          c = atol("123456789");
14          x = a * 3;
15          y = b * 3.0;
16          z = c * 3;
17          printf("%d¥n", x);
18          printf("%lf¥n", y);
19          printf("%ld¥n", z);
20      }
```

〔３〕文字操作関数

例題１－２１

出力結果１－２１のように，JIS コード 0～127 までの中から 16 進数表示文字,英小文字,英大文字をすべて出力するプログラムをつくりなさい。

▼出力結果１－２１

```
16 進数表示文字 = 0123456789ABCDEFabcdef
英小文字 = abcdefghijklmnopqrstuvwxyz
英大文字 = ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

考え方

文字が 16 進数表示なら真となる isxdigit 関数,英小文字なら真となる islower 関数,英大文字なら真となる isupper 関数を使ってプログラムをつくる。

▼プログラム１－２１

```
01      /* E1-21 */
02      /* 文字種テスト関数 */
03      #include <stdio.h>
04      #include <ctype.h>
05
06      main()
07      {
08          int c;
09
10          printf("¥n16進数表示文字 = ");
11          for (c = 0; c <= 127; c++) {
12              if (isxdigit(c))
13                  printf("%c", c);
14          }
15
16          printf("¥n英小文字 = ");
17          for (c = 0; c <= 127; c++) {
18              if (islower(c))
19                  printf("%c", c);
20          }
21
22          printf("¥n英大文字 = ");
23          for (c = 0; c <= 127; c++) {
24              if (isupper(c))
25                  printf("%c", c);
26          }
27      }
```

練習問題１５

１５-１．JIS コード 0～127 までの中から英数字，英文字，数字，区切り文字をすべて出力するプログラムをつくりなさい。(15-１の提出の必要はありませんが演習してください)
《ヒント》<ctype.h>に入っている文字種のテスト関数を調べてみなさい。

１５－２．（難問）文字列変換関数 atof を使わずに，キーボードから getchar 関数を用いて入力した数字の文字列を浮動小数点数に変換（数値化）してから，その数の平方根を数学関数 sqrt で計算して出力するプログラムをつくりなさい。ただし，キーボードから入力する数字文字列は，小数点を含んでよく，改行で入力を完了するものとする。さらに，負数の入力や誤った表記（例えば，-1.5，23..12，00，34a2b）をチェックできるようにしてみましょう。また，123. や .015 のような小数点の使い方は受け付けるものとします。

▼出力例

正の実数の平方根を計算します！
数を入力して下さい：+123.↵　　<--　先頭の+記号や最後の小数点は受け付ける
入力数は 123.000000 です。
平方根は 11.090537 です。

正の実数の平方根を計算します！
数を入力して下さい：23..12↵　　<--　小数点 2 個は誤り
入力文字列が正しい数字表記でありません！

正の実数の平方根を計算します！
数を入力して下さい：00↵　　<--　先頭に 0 が続くと誤り
入力文字列が正しい数字表記でありません！

正の実数の平方根を計算します！
数を入力して下さい：-1.5↵
入力数は-1.500000 です。
負数の平方根は計算できません！

《ヒント》次のように数字文字列を入力する。
while ((c = getchar()) != '¥n') {
c が数字文字であれば，c - '0' で数に変換できる。