情報処理論2a

1章「最大／最小／平均」

中島康彦

§1. 1 今日の作業ディレクトリを作る

1．% cd ⇒ ホームディレクトリへ移動

2．% mkdir chap13 ⇒ ディレクトリchap13を作成

3．% cd chap13 ⇒ ディレクトリchap13へ移動

4．% netscapeを使ってdata13をchap13へダウンロード

5．% tar xvf data13 ⇒ サンプルデータの複写

```
stat1.c
stat2.c
```

§1. 2 最大／最小／算術平均／幾何平均／調和平均

機能設計(入出力形式)

標準入力から整数を読み込む.
読み込んだ整数の総数をnとし，以下を表示する.

最大	… max(D1, D2, ..., Dn)
最小	… min(D1, D2, ..., Dn)
算術平均	… (D1 + D2 + ... + Dn)/n
幾何平均	… $\sqrt[n]{(D1 * D2 * ... * Dn)}$
調和平均	… n/(1/D1+1/D2+ ... +1/Dn)

構成設計(内部データ構造)

総数を記憶する変数	… int型 n
最大値を記憶する変数	… int型 max
最小値を記憶する変数	… int型 min
算術平均を記憶する変数	… double型 arithmetic_mean
幾何平均を記憶する変数	… double型 geometric_mean
調和平均を記憶する変数	… double型 harmonic_mean

§1. 3 最大=max(D1, D2, ..., Dn)

- 8ビット符号付き整数(char型)

```
10000000 … -128(最小値)
11111111 … -1
00000000 …  0
```

- 4バイト符号付き整数(int型)の最小値

```
10000000000000000000000000000000 … 0x80000000
```

- 変数maxの初期値を表現可能な最小値とし，より大きなデータ値に置き換えていく.

```
int max=1<<(sizeof(int)*8-1);   /*31bit左シフト*/
int data;                       /*読み込みデータの格納先*/
while (読み込みデータ⇒data)
  if (max < data) max = data;
```

- 最大値は，maxに残る.

§1. 4 最小=min(D1, D2, ..., Dn)

- 8ビット符号付き整数(char型)

```
00000000 ...  0
00000001 ... +1
01111111 ... +127(最大値)
```

- 4バイト符号付き整数(int型)の最大値

```
01111111111111111111111111111111 ... 0x7fffffff
```

- 変数minの初期値を表現可能な最大値とし，より小さなデータ値に置き換えていく.

```
int min = ~max;                /* maxのビット反転 */
int data;                  /* 読み込みデータの格納先 */
while (読み込みデータ⇒data)
  if (min > data) min = data;
```

- 最小値は，minに残る.

§1. 5 演算に関する留意点

C言語は整数桁溢れ例外を検出しない.

- 整数最大値+1は整数最小値として続行される.
- 最大値に近い整数や多数の整数を加算した場合，正しい結果かどうかがわからない.
- このような場合は，結果を浮動小数点数に蓄積する.

int型の加算結果は，より有効数字の多いdouble型へ蓄積する. 有効数字の少ないfloat型は使えない.

```
int型     ...符号1ビット，有効数字31ビット
          2147479552+1          ⇒ 2147479553
double型 ...符号1ビット，指数部11ビット，有効数字52ビット
          2147479552.0+1.0      ⇒ 2147479553.0
float型   ...符号1ビット，指数部8ビット，有効数字23ビット
          2147479552.0+1.0      ⇒ 2147479552.0 のまま
```

§1. 5 演算に関する留意点(続き)

整数除算の場合，小数点以下は切り捨てられる.
異なる型の間で演算を行う場合には，明示的に型変換を行う習慣をつけるべきである.

- 一般に，整数のみを含む演算は`int`型，浮動小数点数を含む演算は`double`型として行われる.
- 必要に応じて自動的に型変換が行われるが，誤解や思い込みがあると，問題究明に時間がかかる.

```
int i = 3;
int j = 2/i*10;        /* jは0 */
int k = 2.0/i*10;      /* kは6 */

int k = (int)(2.0/(double)i*10.0) と記述すべき.
```

§1. 6 算術平均=(D1 + D2 + ... + Dn)/n

- 読み込みデータ数を累算する.
- 同時にデータを加算する.

```
int data;              /* 読み込みデータの格納先 */
int n = 0;                 /* データ数を記憶する */
double arithmetic_mean = 0.0;  /* 初期値0.0 */

while (読み込みデータ⇒data) {
  arithmetic_mean += (double)data;/* 型変換 */
  n++;                          /* nを1増やす */
}
```

- 算術平均は，`arithmetic_mean/(double)n`

§1. 7 幾何平均=n√(D1 * D2 * ... * Dn)

- 読み込みデータ数を累算する.
- 同時にデータを乗算する.

```
#include <math.h>  /*関数pow()の定義が入っている*/
int data;              /* 読み込みデータの格納先 */
int n = 0;                 /* データ数を記憶する */
double geometric_mean = 1.0;    /* 初期値1.0 */

while (読み込みデータ⇒data) {
  geometric_mean *= (double)data; /* 型変換 */
  n++;                          /* nを1増やす */
}
```

- 幾何平均は, doubleを返す関数pow(a,b)を使うと
  `pow(geometric_mean, 1.0/(double)n)`

§1. 8 調和平均=n/(1/D1 + 1/D2 + ... + 1/Dn)

- 読み込みデータ数を累算する.
- 同時にデータの逆数を加算する.

```
int data;              /* 読み込みデータの格納先 */
int n = 0;                 /* データ数を記憶する */
double harmonic_mean = 0.0;     /* 初期値0.0 */

while (読み込みデータ⇒data) {
  harmonic_mean += 1.0/(double)data;/*型変換*/
  n++;                              /*nを1増やす*/
}
```

- 調和平均は, `(double)n/harmonic_mean`

§1. 9 関数呼出しに関する留意点

- 関数からの返り値は，通常，多くても1つである.
- `main()`からの返り値は，シェルに返る.
- `exit(a)`はプログラムを終了し，aをシェルに返す.
- 以下の例では，`int`型を返り値とする関数`sub(a,b)`において，aとbの値を変更している.
- しかし，jとkに格納される値には，影響を与えない.

```
main() {
  int a=1, b=2, i, j, k;
  i = sub(a, b);        /* 返り値5がiに格納される */
  j = a; k = b;         /* jに1, kに2が格納される */
}
int sub(int a, int b) {
  a = 3;                /* 引数a,bの値を変更しても */
  b = 4;                /* 呼び出し元には影響しない */
  return(5);                        /* 返り値は5 */
}
```

§1. 9 関数呼出しに関する留意点(続き)

- 複数の返り値を必要とする場合，引数に格納させる.
- 呼び出し側は&演算子により変数のアドレスを渡す.
- 関数側は*演算子を使ってアドレスに直接書き込む.
- アドレスを保持する変数をポインタと呼ぶ.

```
main() {
  int a=1, b=2, i, j, k;
  i = sub(&a, &b);      /* 返り値5がiに格納される */
  j = a; k = b;         /* jに3, kに4が格納される */
}

int sub(int *a, int *b) {
  *a = 3;                   /* a,bはポインタであり */
  *b = 4;                   /* アドレスを保持している */
  return(5);                        /* 返り値は5 */
}
```

§1. 10 標準入力からのデータ読み込み

関数scanf() … 代入に成功した個数を返す.

- 第1引数の書式に従い第2引数以降に順次格納する.
- すなわち, 第2引数以降はポインタである.
- scanf("%d", &data);は, 数字(文字列)をint型の数値に変換し, 変数dataに格納する.
- 第1引数の書式. intの代わりにunsigned等でもOK.
  - %d … 10進数とみなしintに変換. 引数はintへのポインタ.
  - %o … 8進数とみなしintに変換. 同上.
  - %x … 16進数とみなしintに変換. 同上.
  - %f … floatに変換. 対応引数はfloatへのポインタ.
  - %lf… doubleに変換. 対応引数はdoubleへのポインタ.
  - %c … 1文字とみなし, 対応引数(charへのポインタ)へ格納.
  - %s … 文字列とみなし, 対応引数(charへのポインタ)の先頭から順に格納.

§1. 11 標準出力へのデータ書き出し

関数printf()

- 第1引数の書式に従い第2引数以降を順次出力する.
- 以下の例では, 各々の型の数値を文字列に変換して, 出力する.

```
printf("n=%d max=%d min=%d\n", n, max, min);
```

- 第1引数の書式. intの代わりにunsigned等でもOK.
  - %d … 符号付き10進数として出力. 引数はint.
  - %u … 符号無し10進数として出力. 引数はint.
  - %o … 符号無し8進数として出力. 引数はint.
  - %x … 符号無し16進数として出力. 引数はint.
  - %e … [-]d.ddde+-ddとして出力. 引数はdouble.
  - %f … [-]ddd.dddとして出力. 引数はdouble.
  - %g … 指数の大きさに応じて%fまたは%eにて出力.
  - %c … 1文字として出力. 引数はint.
  - %s … 文字列として出力. 引数はcharへのポインタ.
  - %% … %文字を出力.

§1. 12 プログラムにまとめる(stat1.c)

```
static char RcsId[] = "$Header$"; /* RCSを使うと日付/版数/作者などに置き換わる */
#include <stdio.h>
#include <math.h>
main()
{
  int data;
  int n = 0;
  int max = 1<<(sizeof(int)*8-1), min = ~max;
  double arithmetic_mean = 0.0;
  double geometric_mean  = 1.0;
  double harmonic_mean   = 0.0;

  while (scanf("%d", &data) == 1) { /* 数字以外を入力すると終了する */
    if (max < data) max = data;
    if (min > data) min = data;
    arithmetic_mean += (double)data;
    geometric_mean  *= (double)data;
    harmonic_mean   += 1.0/(double)data;
    n++;
  }
  printf("n=%d max=%d min=%d\n", n, max, min);
  printf("arithmetic_mean=%g\n", arithmetic_mean/(double)n);
  printf("geometric_mean =%g\n", pow(geometric_mean, 1.0/(double)n));
  printf("harmonic_mean  =%g\n", (double)n/harmonic_mean);
  exit(0);
}
```

§1. 13 入力データに0があると異常終了

0除算例外(Floating exception)が発生し得る.
入力データ中の0を検出して調和平均を止める.

- 制御変数の追加

```
int disable_harmonic_mean = 0;
```

- 0検出の追加

```
if (!data)
  disable_harmonic_mean = 1;
if (!disable_harmonic_mean)
  harmonic_mean += 1.0/(double)data;
```

- レポートの追加

```
if (!disable_harmonic_mean)
  printf("harmonic_mean  =%g\n", ...
else
  printf("harmonic_mean  =N.A.\n");
```

§1. 14 改良版(stat2.c)

```
static char RcsId[] = "$Header$";

#include <stdio.h>
#include <math.h>

main()
{
  int data;
  int disable_harmonic_mean = 0;
  int n = 0;
  int max = 1<<(sizeof(int)*8-1), min = ~max;
  double arithmetic_mean = 0.0;
  double geometric_mean  = 1.0;
  double harmonic_mean   = 0.0;

  while (scanf("%d", &data) == 1) {
    if (max < data) max = data;
    if (min > data) min = data;
    arithmetic_mean += (double)data;
    geometric_mean  *= (double)data;
    if (!data)
      disable_harmonic_mean = 1;
    if (!disable_harmonic_mean)
      harmonic_mean += 1.0/(double)data;
    n++;
  }

  printf("n=%d max=%d min=%d\n", n, max, min);
  printf("arithmetic_mean=%g\n", arithmetic_mean/(double)n);
  printf("geometric_mean =%g\n", pow(geometric_mean, 1.0/(double)n));
  if (!disable_harmonic_mean)
    printf("harmonic_mean  =%g\n", (double)n/harmonic_mean);
  else
    printf("harmonic_mean  =N.A.\n");

  exit(0);
}
```

§1. 15 コンパイルと実行

1. コンパイルする.

```
% gcc stat2.c -o stat2
/var/tmp/ccFy10171.o: In function `main':
/var/tmp/ccFy10171.o(.text+0x10c):
                         undefined reference to `pow'
```

2. 数学関数を使うには，ライブラリをリンクしなければならない.

```
% gcc stat2.c -o stat2 -lm
```

3. 実行

```
% ./stat2
1 2 3 4 5 6 7 8 9 10
e
n=10 max=10 min=1
arithmetic_mean=5.5
geometric_mean =4.52873
harmonic_mean  =3.41417
```

§1. 16 例題

実は，stat2.cの対処は不十分である．stat2.cを異常終了させる入力データを考え，以下をレポートせよ．

- 異常終了させる入力データ
- 実行結果
- 異常終了の理由
- 問題箇所(stat2.cのどの部分であるか)

§1. 17 今日の課題

全入力データが4294967296の時，平均値は0になる．全入力データが2147483648の時，平均値は-2147483648になる．この理由を述べよ．また，./stat2で入力可能な正しいデータの上限値は何かを示せ．

宛先: nakashim@econ.kyoto-u.ac.jp
件名: unix2-学生番号

今日はここまで