プログラミング演習 第 7 回

問 A [07A]：最大値と最小値の同時探索

分割統治法を用いて、整列されていない N 個の int 型の要素を格納した配列の中から最大値と最小値を同時に求める再帰関数 *maxmin* を作成せよ。最大値と最小値の組は次の構造体 MMpair で扱い、*maxmin* のプロトタイプは以下のようになる。

```
typedef struct {int max; int min;} MMpair;

MMpair maxmin(int data[], int left, int right);
```

maxmin は、与えられた配列 data の、left で示される左端 (配列の先頭に近い方) の添字から right で示される右端 (配列の末尾に近い方) の添字までの要素の中から最小値と最大値を探し、MMpair 型の戻り値として返す。

さらに、標準入力 (端末) からマクロ N で示される個数の整数を配列 (たとえば a) に読み込み、maxmin(a, 0, N-1) を呼び出した結果を印字する *main* 関数を用意してプログラムを完成させよ。N の値は 10 とせよ。ただしより大きな値の N に対しても正しく動作するようにすること。

入出力例

入力 1

```
23
29
84
15
58
19
81
17
48
15
```

入力 1 に対する出力

```
15 84
```

入力 2

```
1
2
3
4
5
6
7
8
9
10
```

入力 2 に対する出力

```
1 10
```

ヒント

最小値と最大値を得るために、配列全体を調べるようなことはしてはいけない。代わりに、配列のうち data[left] から data[right] までの最大値と最小値は、この範囲を二つのグループに分けた (たとえば均等に二分割した) それぞれの最大値、最小値から求めることができることを利用する。

問 B [07B]: 再帰の除去

再帰を用いて n の階乗を求める関数 fact1 と、再帰を用いて x^n(n は正の整数) を求める関数 xn1 を作成せよ。このときどちらも末尾再帰となるようにすること。

fact1 と xn1 を用いて、e^x の値をマクローリン展開[1]から求めるプログラムを作成せよ。マクローリン展開の和の項数は 30 とし、$x = 0.5$ とする。

プログラムの実行時間の計測を行うが、そのままでは実行時間が短すぎるため、正しく時間を計測することができない。そこで、e^x の計算の部分を数十万〜数百万回繰り返し、時間を測定せよ。ただし、繰り返し回数は実行時間 3 秒を超えないように決定すること。時間の測定は、

```
time ./a.out
```

のように実行し、$xx.xxxx$u $y.yyy$s... のように表示される部分の $xx.xxxx$(秒) を用いる。繰り返し回数を大きくし過ぎてプログラムが長時間終了しなくなってしまったときには Ctrl キーを押しながら c を押すことでプログラムを中断することができる。

また、同じ c ソースファイルに fact1 および xn1 から再帰を取り除いた関数 fact2 および xn2 を作成し、これらを用いた時の実行時間を計測し、比較せよ。

さらにコンパイル時に最適化オプション-O2 を用いて (O はアルファベット大文字のオー)、

```
gcc -O2 07b.c
```

とコンパイルした場合の実行時間も併せて比較せよ。

レポートには、

- fact1 と xn1 を使用、最適化オプションなし
- fact2 と xn2 を使用、最適化オプションなし
- fact1 と xn1 を使用、最適化オプションあり
- fact2 と xn2 を使用、最適化オプションあり

の 4 通りについて、実行時間を比較し、結果を考察せよ。

提出するプログラムは、fact1 および xn1 を使用し、fact2 および xn2 は、(使用しないが) そのまま残しておくこと。

出力例

```
1.648721
```

ヒント

末尾再帰で記述した fact1() 関数

```
int fact1(int n, int a){
  if (n <= 1 ) return a;
  return fact(n-1, a*n);
}
```

[1] マクローリン展開は、$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} x^n$ で与えられる。プログラムでは、$f(x) \simeq \sum_{n=0}^{29} \frac{f^{(n)}(0)}{n!} x^n$ として計算せよ。

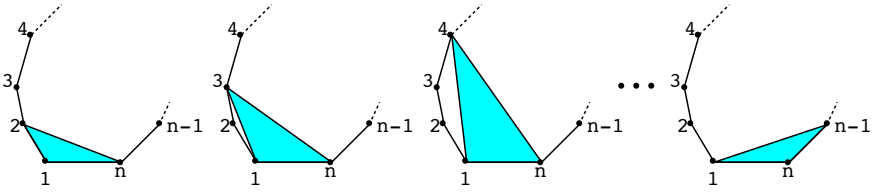

図 1: 凸 n 多角形の三角形への分割の仕方

問 C [07C]: 三角形への分割数

凸 (トツ) n 多角形[2]の頂点を直線で結び互いに重複のない三角形に分割することを考える。異なる分割の仕方の個数 (以下、分割数と呼ぶ) を求める再帰関数 `int divide3(int n)` を作成せよ。たとえば三角形 ($n = 3$) は 1、四角形 ($n = 4$) は 2、五角形 (n=5) は 5 となる。また *divide3* を用いて、標準入力 (端末) から与えられる正整数 $n(> 2)$ に対して `divide3(`n`)` を印字するプログラムを作成せよ。なお、結果は `int` 型に収まるものと仮定してよい。

入出力例

入力 1

```
6
```

入力 1 に対する出力

```
14
```

入力 2

```
9
```

入力 2 に対する出力

```
429
```

ヒント

いずれかの一辺を選び、それと残りの $n - 2$ 頂点を結んでできる三角形を作ると、その三角形を元の凸 n 多角形から除いた部分は、1 つあるいは 2 つの $n - 1$ 頂点以下の凸多角形となる (図 1 参照)。

[2] 異なる任意の二頂点を結んだ線分が全て内部にあるような多角形。

問 D [07D]: 最長一筆書き

鉄道網の駅間距離が行列の形で与えられている。図 2 の例では東京と品川の距離が 7 であることを表している。東京から出発し、この鉄道網を使って同じ区間を高々1 回だけ使って最も長い距離を乗車する経路は、

東京–品川–川崎–横浜–武蔵小杉–大崎–新宿–西国分寺–立川–府中本町–武蔵小杉–川崎

である。この経路を東京から出発する**最長一筆書きの経路**と呼ぶことにする。駅間距離を表す行列が与えられた時、その行列の 1 行目に相当する駅から出発する最長一筆書き経路の長さを出力するプログラムを作れ。図 2 の例では 129 となる。

最長一筆書き経路では、同じ区間を 2 度以上使わなければ、同じ駅を何回通ってもよいし、通らない駅や区間があってもよい。なお、入力では、駅の数は 10 で、区間の距離は整数、経路がない区間は-1 で表わされている。また、行列では一つの区間に対して行きと帰りで同じ距離が 2 回現れるが、同じ区間は高々1 回 (行きか帰りのどちらか) しか通れないことに注意せよ。

入出力例

入力

```
-1  7 -1 -1 -1 -1 10 -1 -1 -1
 7 -1 11 -1 -1  2 -1 -1 -1 -1
-1 11 -1 11  8 -1 -1 -1 -1 -1
-1 -1 11 -1 15 -1 -1 -1 -1 -1
-1 -1  8 15 -1 12 -1 -1 -1 20
-1  2 -1 -1 12 -1  9 -1 -1 -1
10 -1 -1 -1 -1  9 -1 23 -1 -1
-1 -1 -1 -1 -1 -1 23 -1  5  4
-1 -1 -1 -1 -1 -1 -1  5 -1  8
-1 -1 -1 -1 20 -1 -1  4  8 -1
```

出力

```
129
```

ヒント

移動した経路を-1 で置き換えると使ってもよい区間だけが残った新しい行列が得られる。

	A	B	C	D	E	F	G	H	I	J
A. 東京	-	7	-	-	-	-	10	-	-	-
B. 品川	7	-	11	-	-	2	-	-	-	-
C. 川崎	-	11	-	11	8	-	-	-	-	-
D. 横浜	-	-	11	-	15	-	-	-	-	-
E. 武蔵小杉	-	-	8	15	-	12	-	-	-	20
F. 大崎	-	2	-	-	12	-	9	-	-	-
G. 新宿	10	-	-	-	-	9	-	23	-	-
H. 西国分寺	-	-	-	-	-	-	23	-	5	4
I. 立川	-	-	-	-	-	-	-	5	-	8
J. 府中本町	-	-	-	-	20	-	-	4	8	-

図 2: 各駅間の距離の例