第４回日本情報オリンピック 予選問題 解答例

国際情報オリンピック日本委員会

A-1(10 点)

```
#include <stdio.h>
#define N 4

int gcd(int x, int y);

void main(void) {
    int x[N],GCD,y,i;
    unsigned long int LCM;
    /* x[0]〜x[N-1] の最大公約数と最小公倍数を求める */

    for (i=0; i<N; i++) scanf("%d", &x[i]);
    GCD = LCM = x[i=0];
    while (++i<N) {
        y = x[i];
        GCD = gcd(GCD,y);
        LCM = LCM*y/gcd(LCM,y);
    }
    printf("最大公約数は%d、最小公倍数は%d です。\n", GCD,LCM);
}

/* x と y の最大公約数を求める関数（x,y ≧ 0 と仮定する）*/
int gcd(int x, int y) {
    int oldx;
    while (y>0) {
        oldx = x;
        x = y;
        y = oldx % y;
    }
    return x;
}
```

A-2(10 点)

```
#include <stdio.h>

void main(void) {
    char str_A[8], str_B[8];
    int A[8], B[8], C[16];
    int i,j,a,c;

/* １つ目のデータ A の入力
   A[i-1] に第 i 位の桁を格納 */
    printf("１つ目の８桁の正整数 A を入力してください：\n");
    gets(str_A);
    for (i=0; str_A[i]; i++) A[7-i] = str_A[i]-48;

/* ２つ目のデータ B の入力
   B[i-1] に第 i 位の桁を格納 */
    printf("２つ目の８桁の正整数 B を入力してください：\n");
    gets(str_B);
    for (i=0; str_B[i]; i++) B[7-i] = str_B[i]-48;

/* 配列 C の初期化 */
    for (i=0; i<=15; i++) C[i]=0;

/* A * B の実行 */
    for (i=0; i<=7; i++) {
        for (j=0,c=0; j<=7; j++) {
            a = A[j]*B[i] + C[i+j] + c;
            C[i+j] = a%10;
            c = a/10;
        }
        C[j+i] += c;
    }

/* 結果 A*B の出力 */
    printf("A*B = ");
    for (i=15; C[i]==0; i--) ;    /* 上位の 0 をスキップ */
    for ( ; i>=0; i--) printf("%d", C[i]);
    putchar('\n');
}
```

A-3(10 点)

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

#define MAX_NUM   100  /* 入力データの数の上限 */
#define MAX_DATA 2000  /* 入力データの値の上限 */

#define PRIME        1
#define DELETED      0

int sieve[MAX_DATA+1];   /* ふるい用の配列 */

void eratostheness(int max_data);
int prime_pair(int target);

void main(void) {
    FILE *input_file, *output_file;
    int targets[MAX_NUM];
    int n, i, max, prime_1, prime_2;

/* 入力データの読み込み */
    if ((input_file = fopen("INPUT.TXT", "r")) == NULL) {
        printf("INPUT.TXT のオープンに失敗しました。\n");
        exit(1);
    }
    fscanf(input_file, "%d", &n);
    for (i=1,max=0; i<=n; i++) {
        fscanf(input_file, "%d", &targets[i]);
        if (max < (int)targets[i]) max = targets[i];
    }
    fclose(input_file);

/* 素数表の作成 */
    eratostheness(max);

/* 答えの作成と出力 */
    if ((output_file = fopen("OUTPUT.TXT", "w")) == NULL) {
        printf("OUTPUT.TXT のオープンに失敗しました。\n");
        exit(0);
    }
    for (i=1; i<=n; i++) {
        prime_1 = prime_pair(targets[i]);
        prime_2 = targets[i] - prime_1;
        fprintf(output_file, "%d %d\n", prime_1, prime_2);
    }
    fclose(output_file);
}
```

```
/* エラトステネスのふるいを使って、素数表を作る */
void eratostheness(int max_data) {
    int i,n,k,sqrt_of_max_data;

    for (i=2; i<=max_data; i++) sieve[i]=PRIME;
    for (i=4; i<=max_data; i+=2) sieve[i]=DELETED;
    n=3;
    sqrt_of_max_data = (int)sqrt((double)(max_data+1));
    do {
        if (sieve[n] != DELETED)
            for (i=n, k=max_data/n; i<=k; i++)
                sieve[n*i] = DELETED;
    }while ((n+=2) <= sqrt_of_max_data);
}

/* 偶数 target が入力として与えられると,
    i, target-i とも素数となる i を１つ返す */
int prime_pair(int target) {
    int i=2;

    while (sieve[i]!=PRIME || sieve[target-i]!=PRIME)
        i++;
    return(i);
}
```

A-4(15 点)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_L 9

typedef struct shelf {           /* 整理棚 */
    char title[MAX_L];           /* 資料のタイトル */
    struct shelf *next_material; /* 次の資料へのポインタ */
} shelf_type;

/* 新しい資料のための領域の確保 */
shelf_type *material_alloc(void) {
    return (shelf_type *)malloc(sizeof(shelf_type));
}

void main(void) {
    FILE *input_file, *output_file;
    shelf_type *leftmost, /* 一番左の資料へのポインタ */
               *new_material, *current, *target;
    char operation, current_title[MAX_L];
    int i, num;

  /* 本棚の初期化 */
    leftmost = material_alloc();
    leftmost->next_material = NULL;

  /* 入出力ファイルのオープン */
    if ((input_file = fopen("INPUT.TXT", "r")) == NULL) {
        printf("INPUT.TXT のオープンに失敗しました。\n");
        exit(1);
    }
    if ((output_file = fopen("OUTPUT.TXT", "w")) == NULL) {
        printf("OUTPUT.TXT のオープンに失敗しました。\n");
        exit(1);
    }

  /* 資料の整理 */
    while ((operation=fgetc(input_file)) != '.') {
        fgetc(input_file);
        for (i=0; (current_title[i]=fgetc(input_file)) != '\n'; i++) ;
        current_title[i]=NULL; printf("\n");
        printf("OPERATION = %c, TITLE = %s\n", operation, current_title);
        if (operation == 'N') {
          /* 新しい資料を整理棚の一番左に加える */
            new_material = material_alloc();
            new_material->next_material = leftmost->next_material;
            leftmost->next_material=new_material;
```

```
            strcpy(new_material->title, current_title);
            printf("Add the new material: %s\n", new_material->title);
            for (i=1,current=leftmost; current->next_material != NULL; i++) {
                current=current->next_material;
                printf("  %d:%s ",i,current->title);
            } fputc('\n',stdout);
        } else {
          /* 資料を探して、出力ファイルに書き込み、
             整理棚の一番左に置く */
            printf("now finding %s --> ", current_title);
            current = leftmost;
            num=1;
            while (1) {
                if (current->next_material == NULL) {
                  /* 該当する資料がない場合 */
                    fprintf(output_file, "%d\n", num);
                    printf("Can't find the target: num=%d\n", num);
                    break;
                }
                if (strcmp(current->next_material->title,current_title)==0) {
                  /* 該当する資料が見つかった場合 */
                    printf("*HIT*: %d:%s\n", num,current->next_material->title);
                    fprintf(output_file, "%d\n", num);
                    target=current->next_material;
                    current->next_material=target->next_material;
                    target->next_material=leftmost->next_material;
                    leftmost->next_material=target;
                    for (i=1,current=leftmost; current->next_material != NULL; i++) {
                        current=current->next_material;
                        printf("  %d:%s ",i,current->title);
                    } fputc('\n',stdout);
                    break;
                }
                printf("  %d:%s ", num,current->next_material->title);
                current=current->next_material;
                num++;
            }
        }
    }
    fclose(input_file);
    fclose(output_file);
}
```

A-5(15 点)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define LEN 4+1
#define NREF 2
#define MAX 1000

struct {
    char var[LEN];
    char op;
    int ref[NREF];
} form[MAX];

vfind(char key[], int cnt){ int i;
    for (i = 0; i < cnt; i++)
        if (strcmp(form[i].var,key) == 0)
            return i;
    return -1;
}

main(){ int cnt, xref, col[MAX], i, k;
    FILE *f = fopen("input.txt","r");
    for (cnt = 0; !feof(f) && cnt < MAX;){ char x[LEN], y[NREF][LEN], op, i;
        fscanf(f,"%s = %s %c %s\n", x, y[0], &op, y[1]);
        xref = vfind(x,cnt);
        if (xref < 0) strcpy(form[xref = cnt++].var,x);
        form[xref].op = op;
        for (i = 0; i < NREF; ++i)
            if (isalpha(y[i][0])){
                int k = vfind(y[i],cnt);
                if (k < 0) strcpy(form[k = cnt++].var,y[i]);
                form[xref].ref[i] = k;
            } else form[xref].ref[i] = -atoi(y[i]);
    }
    fclose(f);

    for (i = k = 0; i < cnt; ++i)
        if (form[i].op != '\0') col[k++] = i;
    for (cnt = k, i = 0; i < cnt; ++i){ int j;
        for (j = i+1; j < cnt; ++j){ int k;
            for (k = 0; k < NREF; ++k)
                if (form[col[i]].ref[k] == col[j]){
                    int t = col[i]; col[i] = col[j]; col[j] = t;
                    break;
                }
        }
```

```
    }

    f = fopen("output.txt","w");
    for (k = 0; k < cnt; k++){ i = col[k];
        fprintf(f,"%s = ",form[i].var);
        if (form[i].ref[0] < 0) fprintf(f,"%d",-form[i].ref[0]);
        else fprintf(f,"%s",form[form[i].ref[0]].var);
        fprintf(f," %c ",form[i].op);
        if (form[i].ref[1] < 0) fprintf(f,"%d",-form[i].ref[1]);
        else fprintf(f,"%s",form[form[i].ref[1]].var);
        fprintf(f,"\n");
    }
    fclose(f);
}
```

B-1(15 点)

$f_n(x)$ の k 次の係数を $a(n,k)$ とおくと、$f_n(x)=\sum_{k=0}^{\infty} a(n,k)x^k$ と表される。$f_n(0)=a(n,0)$ に注意すると、$f_n(0)-f_{n-2}(0)=0$ より

$$a(n,0)=\begin{cases} 0 & n\text{が偶数のとき} \\ 1 & n\text{が奇数のとき} \end{cases} \tag{i}$$

を得る。

一般に、 $f_n(x)-f_{n-2}(x)=xf_{n-1}(x)$ より、

$$a(n,k)-a(n-2,k)=a(n-1,k-1) \tag{ii}$$

が成り立つ。

(1) n が 3 以上の奇数のとき、(i), (ii) より

$$a(n,1)-a(n-2,1)=a(n-1,0)=0$$

だから、$a(n,1)=a(1,1)=0$ となる。

n が偶数のとき、(i), (ii) より

$$a(n,1)-a(n-2,1)=a(n-1,0)=1$$

であり、 n を 2 から n まで動かして加えると

$$\begin{array}{rlcll} & a(n,1) & - & a(n-2,1) & = & 1 \\ & a(n-2,1) & - & a(n-4,1) & = & 1 \\ & & & & \vdots & \\ & a(4,1) & - & a(2,1) & = & 1 \\ +) & a(2,1) & - & a(0,1) & = & 1 \\ \hline & a(n,1) & - & a(0,1) & = & \sum_{i=1}^{n/2} 1 \end{array}$$

したがって、 $a(0,1)=0$ より $a(n,1)=\dfrac{n}{2}$ となる。

答 $\quad a(n,1)=\begin{cases} \dfrac{n}{2} & n\text{が偶数のとき} \\ 0 & n\text{が奇数のとき} \end{cases}$

(2) n が偶数のとき、(ii) より

$$a(n,2) - a(n-2,2) = a(n-1,1) = 0$$

だから、$a(n,2) = a(0,2) = 0$ となる。

n が 3 以上の奇数のとき、(ii) より

$$a(n,2) - a(n-2,2) = a(n-1,0) = \frac{n-1}{2}$$

であり、 $a(2,1) = 0$ より

$$\begin{aligned} a(n,2) &= \sum_{i=1}^{(n-1)/2} a(2i,1) \\ &= \sum_{i=1}^{(n-1)/2} i \\ &= \frac{1}{2}\left(\frac{n-1}{2}\right)\left(\frac{n-1}{2}+1\right) \\ &= \frac{n^2-1}{8} \end{aligned}$$

となる。

答 $$a(n,2) = \begin{cases} 0 & n\text{が偶数のとき} \\ \dfrac{n^2-1}{8} & n\text{が奇数のとき} \end{cases}$$

B-2(15 点)

比較する数の集合を $\mathrm{Input} = \{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D},\mathrm{E},\mathrm{F},\mathrm{G},\mathrm{H}\}$ と表すことにする。また、$x \in \mathrm{Input}$ に対して、$\alpha(x)$ で既に x より小さいことがわかっている数の個数、$\beta(x)$ で既に x より大きいことがわかっている数の個数を表すものとする。また、比較を行った際に、大きいことがわかった数を勝者、小さいことがわかった数を敗者と呼ぶことにする。

(1) 8 個の数を適当に 4 組に分け比較する。その勝者 4 つを、また 2 組に分け比較する。さらにこの 2 つの勝者を比較した勝者が最大の数とする。また、最初の 4 組の比較での敗者を、適当に 2 組に分け比較し、さらにこの 2 つの敗者を比較し小さかった方を最小の数とする。このように、最大値と最小値を定めれば 10 回の比較で十分である。

$\beta(x) = 0$ である数が 2 つ以上存在する間は、最大の数を決定できない。同様に、$\alpha(x) = 0$ である数が 2 つ以上存在する間は、最小の数を決定できない。1 回の比較で、$\beta(x) = 0$ である数は、高々 1 つしか減らない。同様に、1 回の比較で、$\alpha(x) = 0$ である数は、高々 1 つしか減らない。これらのことから、最大を求めるには 7 回の比較をすることが必要である。この比較において、$beta(0)$ のままである数は高々 4 個である。最小の数を決定するには、さらに 3 回の比較が必要である。このことから、最低でも 1 0 回の比較が必要なことがわかる。

(2) (1) の場合と同様に、最大の数を見つける。その後、最大の数と直接比較して敗者となった 3 つの数で、2 回の比較を行い 2 番目の大きな数を決定できる。よって、9 回の比較で十分である。

$\beta(x) \leq 1$ の数が 3 つ以上ある間は、2 番目に大きな数は決定できない。(1) の場合と同様に、最大の数を見つけるには、7 回の比較が必要である。これらの比較において、$\beta(x) = 0$ の数は 1 つになり、それ以外の数は、$beta(x) \geq 1$ になっている。最大の数以外に負けた数

は、$\beta(x) \geq 2$ になっているので、２番目に大きな数の資格は失っている。今後は、どのような順番で最大の数を決めたとしても、最大の数と直接比較した数は、３個以上となる Input が存在することを示す。このことがいえれば、最大値と２番目に大きな数を決めるのに７回以上の比較が必要なことが分かる。

最大値を決める７回の比較の順序を任意にひとつ固定する。Input の各要素 x, y 間の大小関係を次のように決める。

1. $\beta(x) = 0, \beta(y) \geq 1$ ならば $x \geq y$ とする。
2. $\beta(x) = \beta(y) = 0, \alpha(x) > \alpha(y)$ ならば $x \geq y$ とする。

上記に場合に当てはまらない場合は、矛盾が生じない限りどのように大小関係を定めても良い。

このような大小関係の Input に対して、今固定している順序で比較を行った場合に最大値と定まった数 x が m 回の比較をした時点では、

$$\alpha(x) \leq 2^m - 1$$

となる。

(証明) m に関する帰納法 $(1 \leq m \leq 7)$ で証明する。

- $m = 1$ のとき。明らかに $\alpha(x) = 1$。一方、$2^1 - 1 = 1$ なので、成り立つ。
- $m = k - 1$ のとき成り立つとして、$m = k$ の場合を考える。m 回目の比較は、x と y の間で行われたとする。帰納法の仮定より、$\alpha(x) \leq 2^{k-1} - 1$。また、$x$ は、y に勝ので、$\alpha(y) \leq 2^{k-1} - 1$。よって、

$$\alpha(x) \leq \alpha(x) \leq 2^{k-1} - 1 + \{1 + \alpha(x) \leq 2^{k-1} - 1\} = 2^k - 1$$

となる。

x が１位に決定するには、$\alpha(x) = 7$ にならないとならないので、

$$7 = \alpha(x) \leq 2^m - 1$$

より、$2^m \geq 8$ すなわち、$m \leq 3$ が成り立つ。

B-3(10 点)

ゲームの開始時に、グループ１のカードが無条件で選ばれ２枚に減る。よって、３枚目の１が選ばれた時点でゲームは終わる。つまり、プレーヤーが勝つには、３枚目の１が一番最後に選ばれることが必要である。逆に、ゲームが開始した時点で１以外のグループには、カードが３枚ずつあるので、１以外のカードでゲームが終了することはない。また、このゲームでは、プレーヤーは（カードの裏しか見ることができないので）ゲームが終了するまで結果が分からないが、カードが配られた時点でゲームがどのように進行するか決定している。よって、求める確率は、３０枚のカードを一列に並べた際に、一番最後に１がくる確率

$$\frac{1}{10}$$

と等しくなる。