問題は 4 問ある。それとは別におまけ問題が 2 問ある。

問題 1. n 番目の正四面体数 T_n は

$$T_n = n(n+1)(n+2)/6 \tag{1}$$

と表せることが知られている。

プログラム 1-1 は (1) をそのまま実装したものであるが、これでは微妙なところで不都合が生じる。T_n が ULLONG_MAX 以下であっても $n(n+1)(n+2)$ が ULLONG_MAX を越えれば、計算途中でオーバーフローが起きてしまう。

プログラム 1-2 はその不都合に対処したものである。n を 6 で割った余りに応じて事前に割り算を行うことで、計算途中で ULLONG_MAX を越えることを防いでいる。計算結果が ULLONG_MAX を越えたらどうしようもないが、そこはあきらめている。なお、事前の割り算においては、以下の事実を利用している。

- n を 6 で割った余りが 0 のとき、n は 6 で割り切れる。
- n を 6 で割った余りが 1 のとき、$(n+1)(n+2)$ は 6 で割り切れる。
- n を 6 で割った余りが 2 のとき、$n(n+1)$ と $(n+1)(n+2)$ は 6 で割り切れる。
- n を 6 で割った余りが 3 のとき、$n(n+1)$ は 6 で割り切れる。
- n を 6 で割った余りが 4 のとき、$n+2$ は 6 で割り切れる。
- n を 6 で割った余りが 5 のとき、$n+1$ は 6 で割り切れる。

(a) 空欄を埋めよ。

プログラム 1-1: n 番目の正四面体数を計算する関数（素朴な実装）

```
unsigned long long
tetrahedral_number(unsigned long long n)
{
        return n * (n + 1) * (n + 2) / 6;
}
```

プログラム 1-2: n 番目の正四面体数を計算する関数（ちょっと工夫した実装）

```
unsigned long long
tetrahedral_number(unsigned long long n)
{
        switch (n % 6) {
        case 0:
                return （あ）;
        case 1:
                return （い）;
        case 2:
                return (n * (n + 1) / 6) * (n + 2);
        case 3:
                return （う）;
        case 4:
                return （え）;
        case 5:
                return （お）;
        }
        return 0;               /* makes compiler happy */
}
```

問題 2. $a+bi$（ただし、a と b は整数、i は虚数単位）の形で表せる複素数を**ガウス整数**と呼ぶ。ガウス整数とガウス整数の積もガウス整数となる。積は

$$(a+bi)(p+qi) = (ap-bq)+(aq+bp)i$$

で計算できる。

ガウス整数をプログラム 2-1 のように C の構造体を使って実装するものとする。プログラム 2-2 とプログラム 2-3 はガウス整数の積の計算を実装したものである。両者は同じ計算を行う関数だが、引数の渡し方と返値の返し方が異なる。

(a) 空欄を埋めよ。

プログラム 2-1：ガウス整数の構造体による実装

```
struct Gint {
        long long     re;  /* a + bi の a */
        long long     im;  /* a + bi の b */
};
```

プログラム 2-2：ガウス整数の積

```
//  ab: a + bi
//  pq: p + qi
//  返値: (a + bi)(p + qi)
struct Gint
gmult(struct Gint ab, struct Gint pq)
{
        Gint    result;
        （か） = （き） * （く） - （け） * （こ）;
        （さ） = （し） * （す） + （せ） * （そ）;
        return result;
};
```

プログラム 2-3：ガウス整数の積

```
//  ab: ポインタの指す先に a + bi が格納されている
//  pq: ポインタの指す先に p + qi が格納されている
//  xy: ポインタの指す先に (a + bi)(p + qi) を格納する
void
gmult(const struct Gint *ab, const struct Gint *pq, struct Gint *xy)
{
        （た） = （ち） * （つ） - （て） * （と）;
        （な） = （に） * （ぬ） + （ね） * （の）;
};
```

問題 3. プログラム 3-1 もプログラム 3-2 も int の配列の要素のうち偶数の個数を求める関数の実装である。

(a) 空欄を埋めよ。

プログラム 3-1: 配列の要素のうち偶数を数える

```
#include <stdlib.h>

// array: 配列の先頭
// size: 配列の大きさ
// 返値: 偶数の個数
size_t
count_even(const int *array, size_t size)
{
        size_t  counter = 0;
        size_t  i;
        for (i = （は）; i < （ひ）; i ++) {
                if (（ふ） % 2 == 0) {
                        counter ++;
                }
        }
        return counter;
}
```

プログラム 3-2: 配列の要素のうち偶数を数える

```
#include <stdlib.h>

// array: 配列の先頭
// size: 配列の大きさ
// 返値: 偶数の個数
size_t
count_even(const int *array, size_t size)
{
        size_t  counter = 0;
        const int       *p;
        for (p = （へ）; p < （ほ）; p ++) {
                if (（ま） % 2 == 0) {
                        counter ++;
                }
        }
        return counter;
}
```

問題 4. たくさんの数の相乗平均を計算する C の関数が必要になり、今泉慎太郎と西園寺守は、それぞれ、プログラム 4-1, 4-2 のように書いた。数が多い場合は西園寺守版のほうが安全であるが、その理由を説明せよ。

プログラム 4-1：相乗平均を計算する関数（今泉慎太郎版）

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

double
gmean(const double *a, size_t size)
{
        size_t  i;
        double  prod = 1;
        for (i = 0; i < size; i ++) {
                prod *= a[i];
        }
        return pow(prod, 1.0 / size);
}
```

プログラム 4-2：相乗平均を計算する関数（西園寺守版）

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

double
gmean(const double *a, size_t size)
{
        size_t  i;
        double  sum = 0;
        for (i = 0; i < size; i ++) {
                sum += log(a[i]);
        }
        return exp(sum / size);
}
```

おまけ問題 1 (自信のない人のみ)**.** ガウス整数 $a+bi$ がガウス整数 $p+qi$ を割り切るとは、あるガウス整数 $x+yi$ が存在して $(a+bi)(x+yi)=p+qi$ が成り立つことである。

以下を証明せよ。

(a) ガウス整数 $a+bi$ はガウス整数 $a+bi$ 自身を割り切る。

(b) ガウス整数 $a+bi$ がガウス整数 $m+ni$ を割り切り、ガウス整数 $m+ni$ がガウス整数 $p+qi$ を割り切れば、ガウス整数 $a+bi$ はガウス整数 $p+qi$ を割り切る。

おまけ問題 2 (自信のない人のみ)**.** 自分でプログラミング言語を作るとしたらどんな名前をつけたいか。つけたい名前を記し、その名前の由来も説明せよ。