モンスター単純群の位数

モンスターとは、散在型単純群のうち最大のモンスター群を意味する。その位数は、人類が辿り着いた意味のある数字の最大の数で **8.08 x 10の54乗**。アボガドロ定数でさえ6.02 x 10の23乗、6.022140857(74) × 10^23 mol⁻¹だ。こんな大きな数をPCやタブレットで扱えるか？C言語で計算するには、どういうデータの型を用いたら良いか？そもそも、手元にあるPCやタブレットでどこまで計算できるのか検証してみよう！

```
/* 2^46 · 3^20 · 5^9 · 7^6 · 11^2 · 13^3 · 17 · 19 · 23 · 29 · 31 · 41 · 47 · 59 · 71
= 808,017,424,794,512,875,886,459,904,961,710,757,005,754,368,000,000,000
≈ 8×10^53. */

#include <iostream>
#include <math.h>
using namespace std;
#define MONSTER 8080174247945128758864599049617107570057543680000000000.0
int main() {
//キャラクターで表示

printf("808,017,424,794,512,875,886,459,904,961,710,757,005,754,368,000,000,000\n");
//有効桁数チェック

long double monst1 = MONSTER;
printf("monst1 = %.1Lf\n\n", monst1);

//19桁まで計算値符合した


// それでは、各項の計算をしていくと、

/*  long double  でのべき乗関数 powl  を使用してみよう*/

//p2^46
long double ldp2 = 2.0L, ld_resultp2;
    ld_resultp2 = powl(ldp2, 46.0L);
    printf("p2: %.1Lf\n", ld_resultp2);
    //70,368,744,177,664

//p3^20
long double ldp3 = 3.0L, ld_resultp3;
    ld_resultp3 = powl(ldp3, 20.0L);
    printf("p3: %.1Lf\n", ld_resultp3);
    //3,486,784,401

//p5^9
long double ldp5 = 5.0L, ld_resultp5;
    ld_resultp5 = powl(ldp5, 9.0L);
    printf("p5: %.1Lf\n", ld_resultp5);
```

```
    //1,953,125


//p7^6
long double ldp7 = 7.0L, ld_resultp7;
    ld_resultp7 = powl(ldp7, 6.0L);
    printf("p7: %.1Lf\n", ld_resultp7);
    //117,649


//p11^2
long double ldp11 = 11.0L, ld_resultp11;
    ld_resultp11 = powl(ldp11, 2.0L);
    printf("p11: %.1Lf\n", ld_resultp11);
    //121


//p13^3
long double ldp13 = 13.0L, ld_resultp13;
    ld_resultp13 = powl(ldp13, 3.0L);
    printf("p13: %.1Lf\n", ld_resultp13);
    //2,197

//p17
printf("\n");
long double p17 = 17.0L;
printf("p17 %.1Lf\n", p17);


long double p19 = 19.0L; printf("p19 %.1Lf\n", p19);
long double p23 = 23.0L; printf("p23 %.1Lf\n", p23);
long double p29 = 29.0L; printf("p29 %.1Lf\n", p29);
long double p31 = 31.0L; printf("p31 %.1Lf\n", p31);
long double p41 = 41.0L; printf("p41 %.1Lf\n", p41);
long double p47 = 47.0L; printf("p47 %.1Lf\n", p47);
long double p59 = 59.0L; printf("p59 %.1Lf\n", p59);
long double p71 = 71.0L; printf("p71 %.1Lf\n\n", p71);


long double a17_71  = 17.0L * 19.0L * 23.0L * 29.0L * 31.0L * 41.0L * 47.0L * 59.0L * 71.0L;
printf("a17_71 = %.1Lf \n", a17_71);

long double a7_13 = ld_resultp7 * ld_resultp11 * ld_resultp13;

printf("a7_13 = %.1Lf \n", a7_13);
//
long double a2_5 = ld_resultp2 * ld_resultp3 * ld_resultp5;
printf("a2_5 = %.1Lf \n", a2_5);
```

```
//
long double monster_result;
monster_result = ld_resultp2 * ld_resultp3 * ld_resultp5 * ld_resultp7 * ld_resultp11 *
ld_resultp13 * p17 * p19 * p23 * p29 * p31 * p41 * p47 * p59 * p71;
printf("\nmonster_result = %.1Lf \n", monster_result);

}
```

808,017,424,794,512,875,886,459,904,961,710,757,005,754,368,000,000,000
monst1 = 8080174247945128493134858876589873302653568405586575360.0

p2: 70368744177664.0
p3: 3486784401.0
p5: 1953125.0
p7: 117649.0
p11: 121.0
p13: 2197.0

p17 17.0
p19 19.0
p23 23.0
p29 29.0
p31 31.0
p41 41.0
p47 47.0
p59 59.0
p71 71.0

// debugging
a17_71 = 53911588082213.0
a7_13 = 31275457213.0
a2_5 = 47921999905593439027200000000.0

monster_result =
8080174247945128758149690536207475462850101048980275200.0
The true data =
8080174247945128758645990496171075700575436800000000
真値比較すると下線16桁〜19桁まで一致

したがって、long double より大きな桁数を定義できるデータ型が必要。または、別の言語、アルゴリズムを採用しなくては。その解決策は、また、別途示そう。

草房誠二郎