ProjectEuler 题目选讲

毕克

清华大学 交叉信息研究院

2014 年 5 月 2 日

ProjectEuler

所有题目均为提交答案，不限制语言，保证每个题目均有在普通计算机上运行 1 分钟的解法。

所以一般来说，不需要考虑时间效率，或者高精度，我们使用 Python。

需要考虑效率，使用 C++。

积分或数学题，使用 Mathematica。

可以根据题目数量或者 scores 进行排序。

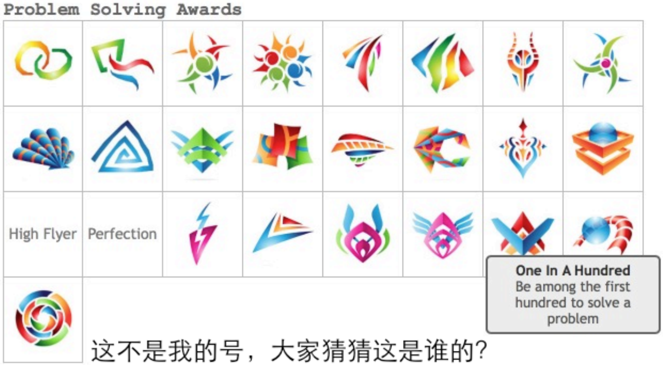
scores 计算方式：

- 每一题前 50 个通过的人有分数，第 i 个人有 $51 - i$ 分。
- 只计算最近 10 道题目的分数中的最高 5 个。

每周末会增加一道题。

	Username	Country	Score	Performance	Language
1st	Anton_Lunyov		247		C/C++
2nd	Nabb		245		Python
3rd	uwi		244		Java
4th	*apia* 这个ID好像在哪见过?		240		Python
5th	Lucy_Hedgehog		233		Python
6th	x22		227		Java

PE 还有自身的成就系统



这不是我的号，大家猜猜这是谁的?

前 100 个通过的可以解锁一个成就。

ProjectEuler 并不鼓励大家写题解，认为这个会让解题变得毫无乐趣，所以本次选题均是通过 100 人以上的水题。

为了更好的阅读/听课/考试体验。
建议大家学习

- $O(n)$ 求 1 到 n 乘法逆元，筛法。
- 微积分相关知识，比如如何用积分计算体积。
- Python，可以节约检查 Overflow 的时间。

发礼物

发礼物

有 n 个球摆成一列，对于任意一个球，都远小于它下面的球的质量。
现在这 n 个球从高度 h 下落，问最后最小的球可以反弹多高。
小球之间均是弹性碰撞。

发另一个礼物

定义一个数为 S 数当且仅当这个数没有大于 3 的质因数。设 $S(N)$ 为所有小于 N 的 S 数，设 $F(N)$ 表示 $S(N)$ 的排列数满足以下条件，每个数在自己所有约数之后出现。

如果 $2^i3^j \le N$，那么选出 (i,j) 这个点。
现在的问题相当于，向这些点填不重复的数，要求 (i,j) 比 $(i-1,j)$，$(i,j-1)$ 都大。
这个问题相当于问 Young tableau（杨氏图表）的个数。
有一个现成的公式 Hook length formula（钩子公式）。
然后只需要一些简单的实现就可以了，我不知道为什么这个题为什么通过人数这么少。

在离地面 100 米的地方发生了爆炸，爆炸产生了成吨的碎片，他们向各个方向飞去，所有都具有 20 米每秒的速度。

假定没有空气阻力，$g = 9.81m/s^2$。

计算碎片经过的体积。

精确到小数点后四位，截断取整。

我们发现在各个方向上一样，我们考虑在 x, z 平面上的情况。
设抛射角度为 θ，轨迹的参数方程是。

$$\begin{cases} x & = vt\cos\theta \\ z & = vt\sin\theta - \frac{1}{2}gt^2 \end{cases}$$

消去 t，可得

$$z = x\tan\theta + \frac{gx^2}{2v^2\cos^2\theta}.$$

$$z = x\tan\theta + \frac{gx^2}{2v^2\cos^2\theta}.$$

设 $k = \tan\theta, k \in \mathbb{R}$

$$z = kx + \frac{gx^2}{2v^2}(1+k^2)$$

变成标准形式

$$k^2 + \frac{2v^2}{gx}k + \left(1 + \frac{2v^2 z}{gx^2}\right) = 0.$$

对于一个确定的 (x, z) 点，需保证 k 有解。

$$\Delta = \left(\frac{2v^2}{gx}\right)^2 - 4\left(1 + \frac{2v^2 y}{gx^2}\right) > 0$$

$$z < \frac{v^4 - g^2x^2}{2gv^2}$$

这是在 x, z 平面上的情况。

解出来 $y = -100$ 时的 x_0。

用柱坐标系计算体积。

$$2\pi \int_0^{x_0} zx \, \mathrm{d}x$$

```
In[178]:= Solve[(v^4 - g^2 x^2) / (2 g v^2) == -100, x] /. {v → 20, g → 981 / 100}
```

Out[178]= $\left\{\left\{x \to -\frac{2000\sqrt{2362}}{981}\right\}, \left\{x \to \frac{2000\sqrt{2362}}{981}\right\}\right\}$

In[179]:= $\texttt{Integrate}\left[x\,((v\verb|^|4 - g\verb|^|2\,x\verb|^|2)/(2\,g\,v\verb|^|2) + 100),\ \left\{x, 0, \frac{2000\sqrt{2362}}{981}\right\}\right] * (2\,\texttt{Pi})$ /. {v → 20, g → 981 / 100}

Out[179]= $\frac{557904400000000\,\pi}{944076141}$

```
In[180]:= N[%, 20]
```

Out[180]= $1.8565328455275742879 \times 10^{6}$

Mathematica 在计算过程中会灵活的控制精度，如果输入 9.81 会被认为是一个近似数，所以为了精确计算可以指定精度或直接输入精确数。

B 君要送给 R 君一个椭球形的糖果，具体来说这个糖果的方程为

$$b^2x^2 + b^2y^2 + a^2z^2 = a^2b^2$$

B 君要在这个糖果上糊上厚度为 1 的巧克力，问当 $a = 3, b = 1$ 需要多少巧克力？精确到小数点后 8 位。

首先我们要明白“厚度为 1”的含义，精确的来说应该是对于任意一点的法线方向，巧克力的厚度为 1，所以糊上巧克力之后，不是一个椭球型。注意到 x 轴和 y 轴半长轴相等，我们只需要分析 x, z 平面上的情况就可以了。

$$\begin{cases} x &= a\cos\theta \\ z &= b\sin\theta \end{cases}$$

对于 θ 时，法线方向应为 $(\frac{\mathrm{d}x}{\mathrm{d}\theta}, \frac{\mathrm{d}z}{\mathrm{d}\theta})$。
切线方向为 $(\frac{\mathrm{d}z}{\mathrm{d}\theta}, -\frac{\mathrm{d}x}{\mathrm{d}\theta})$，即是 $(b\cos\theta, a\sin\theta)$，所以加上厚度为 1 的外壳后。方程为

$$\begin{cases} x &= a\cos\theta + \frac{b\cos\theta}{\sqrt{a^2\sin^2\theta + b^2\cos^2}} \\ z &= b\sin\theta + \frac{a\sin\theta}{\sqrt{a^2\sin^2\theta + b^2\cos^2}} \end{cases}$$

所以总体积是

$$
\begin{aligned}
&\int_{-b-1}^{b+1} \pi x^2 \, \mathrm{d}z \\
= &\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \pi x^2 \frac{\partial z}{\partial \theta} \, \mathrm{d}\theta
\end{aligned}
$$

而初始的椭球的体积是 $\frac{4}{3}\pi a^2 b$。
如此简单的计算就交给暴算大师 Mathematica 就可以了。

In[218]:=
```
x = a Cos[θ] + b Cos[θ] / Sqrt[a^2 Sin[θ]^2 + b^2 Cos[θ]^2]
```
Out[218]= $3\,\mathrm{Cos}[\theta] + \dfrac{\mathrm{Cos}[\theta]}{\sqrt{\mathrm{Cos}[\theta]^2 + 9\,\mathrm{Sin}[\theta]^2}}$

In[219]:=
```
y = b Sin[θ] + a Sin[θ] / Sqrt[a^2 Sin[θ]^2 + b^2 Cos[θ]^2]
```
Out[219]= $\mathrm{Sin}[\theta] + \dfrac{3\,\mathrm{Sin}[\theta]}{\sqrt{\mathrm{Cos}[\theta]^2 + 9\,\mathrm{Sin}[\theta]^2}}$

In[220]:=
```
NIntegrate[Pi x ^2 D[y, θ], {θ, -Pi / 2, Pi / 2}, WorkingPrecision → 30,
  PrecisionGoal → 20] - 4 / 3 * a^2 b Pi
```
Out[220]= 103.378700960229065329051385396

考虑所有整点 $(a, b, c), 0 \le a, b, c \le n$。
从原点连向所有点，设有 $D(n)$ 条不同的直线。
求 $D(10^{10})$，给出答案的前 9 位和后 9 位。

$$\sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=0}^{n}[\gcd(i,j,k)=1]$$

我们都知道 $[p=1]=\sum_{l|p}\mu(l)$。所以上面的式子变为

$$\sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=0}^{n}\sum_{l|i,l|j,l|k}\mu(l)$$

$$\sum_{l}\left\lfloor\frac{n}{l}\right\rfloor^{3}\mu(l)$$

前一部分可以通过分块，在时间 $O(\sqrt{n})$ 的时间内处理，现在的主要问题是求 $\mu(l)$ 的前缀和。
n 的范围是 10^{10}，之前经典的筛法就不奏效了。

设 $F(n)$ 为 $f(n)$ 的前缀和，$G(n)$ 为 $g(n)$ 的前缀和。满足 $g(n) = \sum_{i|n} f(i)$

$$
\begin{aligned}
G(n) &= \sum_{i=1}^{n} g(i) \\
&= \sum_{i=1}^{n} \sum_{j|i} f(j) \\
&= \sum_{i=1}^{n} \sum_{ji \le n} f(j) \\
&= \sum_{i=1}^{n} F\left(\left\lfloor \frac{n}{i} \right\rfloor\right)
\end{aligned}
$$

注意到 $\left\lfloor \frac{\lfloor \frac{n}{i} \rfloor}{j} \right\rfloor = \left\lfloor \frac{n}{ij} \right\rfloor$ 如果 $G(n)$ 能在 $O(1)$ 的时间内计算出，我们可以暴力算出所有的 $F\left(\lfloor \frac{n}{i} \rfloor\right)$.

$$T(n) = \sum_{i=1}^{\sqrt{n}} \sqrt{i} + \sum_{i=1}^{\sqrt{n}} \sqrt{n/i} = \Theta(n^{\frac{3}{4}})$$

如果我们用一些鬼畜的技巧，用筛法预处理 $O(n^{\frac{2}{3}})$，那么复杂度变为

$$T(n) = \sum_{i=1}^{\sqrt[3]{n}} \sqrt{n/i} = \Theta(n^{\frac{2}{3}}).$$

A Happy Ending.

这样我们可以依次计算 $F\left(\left\lfloor\frac{n}{i}\right\rfloor\right)$。
常见满足条件的 $g(n)$ 和 $f(n)$。

$$[n=1]=\sum_{i|n}\mu(i).$$

$$n=\sum_{i|n}\varphi(i).$$

设

$$S(n,m)=\sum_{i=1}^{m}\varphi(in)$$

求

$$S(510510,10^{11})$$

的后九位。

我们注意到对于和 n 互质的质数 p 有

$$S(np,m) = (p-1)S(n,m) + S(np,m/p)$$

这样可以不断递归，最后只需要计算 $S(1,n)$。
用之前 $\Theta(n^{2/3})$ 的方法即可。

一个六面筛子扔 n 次，设 c 为扔出和上一次结果相同的次数。
比如 $n = 7$，扔出的结果为 $(1, 1, 5, 6, 6, 6, 3)$，那么我们有 3 次扔出的结果和上次相同。所以对于这个结果 $c = 3$。
设 $C(n)$ 为扔出结果的 c 小于 $\pi(n)$ 的个数。
设

$$S(L) = \sum_{1 \le n \le L} C(n).$$

求 $S(50000000) \mod 1000000007$。

考虑 $C(n)$ 的表达式，假设结果中有 k 个位置和之前的相同。

那么这样结果的方案数为

$$\binom{n-1}{k} 6 \cdot 1^k \cdot 5^{n-1-k}$$

所以

$$\begin{aligned} C(n) &= \sum_{k=0}^{\pi(n)} \binom{n-1}{k} 6 \cdot 1^k \cdot 5^{n-1-k} \\ &= \sum_{k=n-1-\pi(n)}^{n-1} \binom{n-1}{k} 6 \cdot 5^k \end{aligned}$$

我们发现 $C(n)$ 不能在小于 $O(\pi(n))$ 的复杂度内计算出。

$$S(L) = \sum_{n=1}^{L} \sum_{k=n-1-\pi(n)}^{n-1} \binom{n-1}{k} 6 \cdot 5^k$$

注意到 k 的上下界随 n 都是增加的，所以我们可以交换一下求和顺序。

$$S(L)=\sum_{k=1}^{L-1} 6\cdot 5^k \sum_{n=p(k)}^{q(k)} \binom{n-1}{k}$$

其中 $p(k)$ 和 $q(k)$ 都是关于 k 的函数。

再利用

$$\binom{n-1}{k}=\binom{n}{k+1}-\binom{n-1}{k+1}$$

可以在预处理之后以 $O(1)$ 的时间解决。

有 n 个椅子排成一个环形，接下来会来一些人随机选择椅子坐下，但是这些人选择有一个原则，那就是绝对不和别人相邻。
设 $E(n)$ 为 n 个椅子情况下，最后空椅子的比例。
求 $E(10^{18})$。

解法：

设 $f(n)$ 为一条线的情况下，期望有多少个椅子被坐。所求即是

$$E(n) = 1 - \frac{f(n-3)+1}{n}$$

所以问题转化为求 $f(n)$，这有一个很显然的动态规划的做法。

$$
\begin{aligned}
nf(n) &= \sum_{i=0}^{n-3}(f(i) + f(n-3-i)) + 2f(n-2) + 1 \\
&= 2\sum_{i=0}^{n-3} f(i) + 2f(n-2) + 1 \\
&= 2\sum_{i=0}^{n-3} f(i) + 2f(n-2) + 1 \\
&= 2\sum_{i=0}^{n-2} f(i) + 1 \\
&= (n-1)f(n-1) + 2f(n-2) + 1
\end{aligned}
$$

接下来考虑 f 的生成函数 F。f 满足

$$nf(n) = (n-1)f(n-1) + 2f(n-2) + 1 (n \geq 2)$$

所以 F 满足

$$F'(x) = xF'(x) + 2xF(x) + \frac{x}{1-x}$$

翻开任何一本高等数学或者数学分析的书，我们可以找到这种方程的解法。
当然也可以直接用 Mathematica 解出来。

$$F(x)=\frac{1-e^{-2x}}{2(x-1)^2}$$

利用

$$1-e^{-2x}=1-\sum_{k=0}^{+\infty}\frac{(-2)^k x^k}{k!}$$

和

$$\frac{1}{(1-x)^2}=\sum_{k=0}^{+\infty}(k+1)x^k$$

我们可以得到第 n 项的系数

$$f(n) = \sum_{k=1}^{n} \frac{(-2)^{k-1}}{k!}(n-k+1).$$

当然也可以直接用 Mathematica 展开，得到同样的结果。

注意到分母有 $k!$，所以收敛极快，所以只需要计算前几十项即可。

设 $E(x_0, y_0)$ 为用欧几里得距离算法求最大公约数所需要的次数。

比如 $E(6, 10) = E(10, 6) + 1 = E(6, 4) + 2 = E(4, 2) + 3 = E(2, 0) + 4 = 4.$

$$S(N) = \sum_{1 \le x, y \le N} E(x, y)$$

求 $S(5 \cdot 10^6)$.

这个题目的暴力时间远大于思考时间。考虑最显然的方法，枚举 (x, y)，计算 $E(x, y)$，时间复杂度是 $O(n^2 \lg n)$。

这个做法虽然 naïve，但是我们注意到了这个做法十分适合并行计算。

但是我使用 `C++11` 的 `thread` 库并没有获得很好的效果。

一个显而易见的结论是 $\forall x < y, E(x, y) = E(y, x) + 1$，利用这个可以大约减少一半的时间。

考虑一共有 N^2 个点，这 N^2 个点的关系构成一个树，所以有一个很显然的直接 DFS 的 $O(N^2)$ 的做法。

这样可以在约一晚上的时间运行出结果。

我们要用 1×2 和 1×1 的格子填满一个 $1 \times n$ 的网格。
其中 1×2 的格子只有一种，1×1 的格子有 10 种。
设这个方案数是 $T(n)$，设

$$S(n) = \sum_{1 \le a,b,c \le n} \gcd(T(c^a), T(c^b))$$

求 $S(2000) \mod 987898789$。

我们显然有

$T(n)=10T(n-1)+T(n-2), T(0)=1, T(1)=10$。

我们都知道通过这个可以在 $O(\lg n)$ 的时间内求出第 n 项。

但是这对于解决这个题目并没有什么意义。

发挥一下想象力。我们有

$$T(n)=101T(n-2)+10T(n-3)$$

诶，这几个数字好像见过，我们猜测

$$T(n)=T(i)T(n-i)+T(i-1)T(n-i-1)$$

证明也很简单，就是数学归纳法。

$$
\begin{aligned}
& T(i)T(n-i) + T(i-1)T(n-i-1) \\
= \; & 10T(i)T(n-i-1) + T(i)T(n-i-2) \\
& + T(i-1)T(n-i-1) \\
= \; & T(i+1)T(n-i-1) + T(i)T(n-i-2)
\end{aligned}
$$

我们联想到 Fibonacci 数的一个性质

$$F_{n+k} = F_k F_{n+1} + F_{k-1} F_n$$

你可以在 Concrete Mathematics 第二版的 294 页，或者维基百科找到他。

正是因为 Fibonacci 数具有这个性质，我们有

$$\gcd(F_x, F_y) = F_{\gcd(x,y)}$$

我们设 $T'(n+1)=T(n)$。
这样我们有

$$T'(n+1)=T'(i+1)T'(n-i+1)+T'(i)T'(n-i)$$

所以我们类似的有

$$S(n)=\sum_{1\leq a,b,c\leq n} T'(\gcd(c^a+1,c^b+1))$$

我们注意到 $\gcd(c^a+1, c^b+1)$ 是 c^d+1 或者 $c \bmod 2+1$。
其中 d 是与 c 无关的常数。
于是我们通过枚举 a, b 算出所有的 d，然后再枚举 c, d 算出所有的 $T'(c^d+1)$。
计算 $T'(c^d+1)$ 时不能暴力矩阵乘法，需要利用计算 $T'(c^{d-1}+1)$ 时的矩阵。
这个题目容易运算溢出的地方很多，而且不需要太高的效率，Python 是一个很好的选择。

$$F_{n+k} = F_k F_{n+1} + F_{k-1} F_n$$

利用这个性质来求 $F_n \mod p$，我们可以得到一个比矩阵乘法常数更优，而且更好写的做法。

注意到

$$F_{2n} = F_n F_{n+1} + (F_{n+1} - F_n) F_n$$

和

$$F_{2n+1} = F_{n+1} F_{n+1} + F_n F_n$$

所以我们可以利用 F_n, F_{n+1} 计算出 F_{2n}, F_{2n+1}.

设 $P(m,n)$ 为 $m \times n$ 乘法表内不同数的个数。
求 $P(64, 10^{16})$.

一个 Naïve 的想法是：

注意到一个整数会出现多次，我们不妨只算在行号最小出现那一次。

这样只需要枚举行号，然后看这样出现的数中，之前出现了哪些。

复杂度 $O(2^m mn)$，注意因为 n 过大，枚举子集过程中限制最小公倍小于 n 并没有多大作用。

我们不妨转换思路，枚举 i，统计在 $(i-1)n < x \leq in$ 的 x 的个数。
这样每次枚举 i 之后，只需要枚举 $\{i, i+1, \ldots, m\}$ 的一个子集。
我们又注意到一个神奇的性质，如果 j 在上述那个集合内，那么 j 的倍数的存在是没有意义的。
这样我们可以把最大 64 的集合，变成最大 32 的集合，然后暴力枚举子集即可。

我用上面的方法运行了 90min 通过了，在通过之后我发现了一个用 Python 写成，只需要运行 3 秒的程序。

做法是只考虑每个数最后一次出现，于是对于当前的每行，需要用之后的来筛去一部分。

然后类似刚才，把一些是另外一些倍数的去掉。

设 $N(i)$ 为最小的 n 满足 $n!$ 是 $(i!)^{1234567890}$ 的倍数。

设 $S(u) = \sum_{i=10}^{u} N(i)$。

求 $S(1000000) \mod 10^{18}$。

对于计算每一个 $N(i)$，可以二分 n，然后检查所有质因数的次数。
通过暴力我们发现，对于 $N(i)$ 的决定最后结果的质因数，要么是 $N(i-1)$ 的，要么是 i 的一个质因数。
于是我们只需要通过筛法，分解每一个数，就可以解决了。

谢谢大家

感觉 杜瑜皓 在我制作 PDF 时提供的帮助，
并友情提供

	User	Country	Score	Performance	Language
1st	Anton_Lunyov		247		C/C++
2nd	*apia*		239		Python
2nd	grechnik		239		C/C++

虽然他并未能参加 APIO。