1. Ugly Number II

Write a program to find the n-th ugly number.

Ugly numbers are **positive numbers** whose prime factors only include 2, 3, 5.

**Example:**

Input: n = 10  
Output: 12  
Explanation: 1, 2, 3, 4, 5, 6, 8, 9, 10, 12 is the sequence of the first 10 ugly numbers.

**Note:**

1. 1 is typically treated as an ugly number.
2. n **does not exceed 1690**.

**解法1**

暴力搜索。假设已经知道了前n个丑数，求第n+1个丑数，则有：

$$
a\_{n+1} = \min\{2\*a\_i, 3\*a\_j, 5\*a\_k\}, \quad \forall i, j, k\in[1, n]\\
s.t\quad a\_{n+1} > a\_n
$$

class Solution {  
public:  
 int nthUglyNumber(int n) {  
 vector<int>u\_n{1};  
 int prime[3] = {2, 3, 5};  
 while(u\_n.size() < n){  
 int flag = 0;  
 int cur\_n = INT\_MAX;  
 for(int i = u\_n.size() - 1; i >= 0; --i){  
 for(int j = 0; j < 3; ++j){  
 if(u\_n[i]\*prime[j] > u\_n.back()){  
 cur\_n = min(cur\_n, u\_n[i]\*prime[j]);  
 }else{  
 flag++;  
 }  
 }  
 if(flag == 3)break;  
 }  
 u\_n.push\_back(cur\_n);  
 }  
 return u\_n.back();  
 }  
};

但是提交会超时。。。

**解法2** 对解法1进行改进。显然丑数数组是有序的，可以用二分查找完成，查找的问题描述为：

寻找第一次出现的满足 的

搜索过程为：对于区间

* ，则满足条件的肯定在中
* ，则满足条件的肯定在中

typedef long long int LL;  
class Solution {  
public:  
 int nthUglyNumber(int n) {  
 vector<LL>u\_n{1};  
 int prime[3] = {2, 3, 5};  
 while(u\_n.size() < n){  
 LL cur\_n = LLONG\_MAX;  
 for(int j = 0; j < 3; ++j){  
 int idx = bin\_search(u\_n, prime[j]);  
 cur\_n = min(cur\_n, prime[j]\*u\_n[idx]);  
 }  
 u\_n.push\_back(cur\_n);  
 }  
 return u\_n.back();  
 }  
 int bin\_search(vector<LL>&nums, int k){  
 int last\_num = nums.back();  
 int l = 0, r = nums.size()-1;  
 while(l < r){  
 int mid = (l + r) / 2;  
 if(nums[mid]\*k > last\_num)r=mid;  
 else l = mid + 1;  
 }  
 return l;  
 }  
};

**解法3** 注意到事实：如果是丑数，则也是丑数

typedef long long int LL;  
class Solution {  
public:  
 int nthUglyNumber(int n) {  
 priority\_queue<LL, vector<LL>, greater<LL>>q;  
 set<LL>s;  
 int prime[3] = {2, 3, 5};  
   
 q.push(1);  
 s.insert(1);  
 LL ans = q.top();  
 for(int i = 0; i < n; ++i){  
 ans = q.top();  
 q.pop();  
 s.erase(ans);  
 for(int j = 0; j < 3; ++j){  
 if(s.find(ans\*prime[j]) == s.end()){  
 q.push(ans\*prime[j]);  
 s.insert(ans\*prime[j]);  
 }  
 }  
 }  
 return ans;  
 }  
};

**解法4** 根据解法三种事实，利用动态规划

class Solution {  
public:  
 int nthUglyNumber(int n) {  
 int pre2 = 0, pre3 = 0, pre5 = 0;  
 int nums[1690];  
 nums[0] = 1;  
 for(int i = 1; i < n; ++i){  
 int ugly = min(nums[pre2]\*2, min(nums[pre3]\*3, nums[pre5]\*5));  
 nums[i] = ugly;  
 if(ugly % 2 == 0)pre2++;  
 if(ugly % 3 == 0)pre3++;  
 if(ugly % 5 == 0)pre5++;  
 }  
 return nums[n-1];  
 }  
};