GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

Ugly Numbers

Ugly numbers are numbers whose only prime factors are 2, 3 or 5. The sequence
1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, …
shows the first 11 ugly numbers. By convention, 1 is included.
Write a program to find and print the 150'th ugly number.

METHOD 1 (Simple)
Thanks to Nedylko Draganov for suggesting this solution.

Algorithm:
Loop for all positive integers until ugly number count is smaller than n, if an integer is ugly than increment ugly number count.

To check if a number is ugly, divide the number by greatest divisible powers of 2, 3 and 5, if the number becomes 1 then it is an ugly number otherwise not.

For example, let us see how to check for 300 is ugly or not. Greatest divisible power of 2 is 4, after dividing 300 by 4 we get 75. Greatest divisible power of 3 is 3, after dividing 75 by 3 we get 25. Greatest divisible power of 5 is 25, after dividing 25 by 25 we get 1. Since we get 1 finally, 300 is ugly number.

Implementation:

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

/*This function divides a by greatest divisible
  power of b*/
int maxDivide(int a, int b)
{
  while (a%b == 0)
   a = a/b;
  return a;
}

/* Function to check if a number is ugly or not */
int isUgly(int no)
{
  no = maxDivide(no, 2);
  no = maxDivide(no, 3);
  no = maxDivide(no, 5);

  return (no == 1)? 1 : 0;
}

/* Function to get the nth ugly number*/
int getNthUglyNo(int n)
{
  int i = 1;
  int count = 1;   /* ugly number count */

  /*Check for all integers untill ugly count
    becomes n*/
  while (n > count)
  {
    i++;
    if (isUgly(i))
      count++;
  }
  return i;
}

/* Driver program to test above functions */
int main()
{
    unsigned no = getNthUglyNo(150);
    printf("150th ugly no. is %d ",  no);
    getchar();
    return 0;
}
```

This method is not time efficient as it checks for all integers until ugly number count becomes n, but space complexity of this method is O(1)

METHOD 2 (Use Dynamic Programming)
Here is a time efficient solution with O(n) extra space. The ugly-number sequence is 1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, …

because every number can only be divided by 2, 3, 5, one way to look at the sequence is to split the sequence to three groups as below:

(1) 1×2, 2×2, 3×2, 4×2, 5×2, …
(2) 1×3, 2×3, 3×3, 4×3, 5×3, …
(3) 1×5, 2×5, 3×5, 4×5, 5×5, …

We can find that every subsequence is the ugly-sequence itself (1, 2, 3, 4, 5, …) multiply 2, 3, 5. Then we use similar merge method as merge sort, to get every ugly number from the three subsequence. Every step we choose the smallest one, and move one step after.

Algorithm:

```
1 Declare an array for ugly numbers:  ugly[150]
2 Initialize first ugly no:  ugly[0] = 1
3 Initialize three array index variables i2, i3, i5 to point to
   1st element of the ugly array:
        i2 = i3 = i5 =0;
4 Initialize 3 choices for the next ugly no:
         next_mulitple_of_2 = ugly[i2]*2;
         next_mulitple_of_3 = ugly[i3]*3
         next_mulitple_of_5 = ugly[i5]*5;
5 Now go in a loop to fill all ugly numbers till 150:
For (i = 1; i < 150; i++ )
{
    /* These small steps are not optimized for good
      readability. Will optimize them in C program */
    next_ugly_no  = Min(next_mulitple_of_2,
                                  next_mulitple_of_3,
                                  next_mulitple_of_5);
    if  (next_ugly_no  == next_mulitple_of_2)
    {
        i2 = i2 + 1;
        next_mulitple_of_2 = ugly[i2]*2;
    }
    if  (next_ugly_no  == next_mulitple_of_3)
    {
        i3 = i3 + 1;
        next_mulitple_of_3 = ugly[i3]*3;
     }
     if  (next_ugly_no  == next_mulitple_of_5)
     {
        i5 = i5 + 1;
        next_mulitple_of_5 = ugly[i5]*5;
     }
     ugly[i] =  next_ugly_no
}/* end of for loop */
```

```
6.return next_ugly_no
```

Example:
Let us see how it works

```
initialize
   ugly[] =  | 1 |
   i2 =  i3 = i5 = 0;

First iteration
   ugly[1] = Min(ugly[i2]*2, ugly[i3]*3, ugly[i5]*5)
            = Min(2, 3, 5)
            = 2
   ugly[] =  | 1 | 2 |
   i2 = 1,  i3 = i5 = 0  (i2 got incremented )

Second iteration
    ugly[2] = Min(ugly[i2]*2, ugly[i3]*3, ugly[i5]*5)
             = Min(4, 3, 5)
             = 3
    ugly[] =  | 1 | 2 | 3 |
    i2 = 1,  i3 =  1, i5 = 0  (i3 got incremented )

Third iteration
    ugly[3] = Min(ugly[i2]*2, ugly[i3]*3, ugly[i5]*5)
             = Min(4, 6, 5)
             = 4
    ugly[] =  | 1 | 2 | 3 |  4 |
    i2 = 2,  i3 =  1, i5 = 0  (i2 got incremented )

Fourth iteration
    ugly[4] = Min(ugly[i2]*2, ugly[i3]*3, ugly[i5]*5)
              = Min(6, 6, 5)
              = 5
    ugly[] =  | 1 | 2 | 3 |  4 | 5 |
    i2 = 2,  i3 =  1, i5 = 1  (i5 got incremented )

Fifth iteration
    ugly[4] = Min(ugly[i2]*2, ugly[i3]*3, ugly[i5]*5)
              = Min(6, 6, 10)
              = 6
    ugly[] =  | 1 | 2 | 3 |  4 | 5 | 6 |
    i2 = 3,  i3 =  2, i5 = 1  (i2 and i3 got incremented )

Will continue same way till I < 150
```

Program:

```
# include<stdio.h>
# include<stdlib.h>
# define bool int

/* Function to find minimum of 3 numbers */
unsigned min(unsigned , unsigned , unsigned );

/* Function to get the nth ugly number*/
unsigned getNthUglyNo(unsigned n)
{
    unsigned *ugly =
             (unsigned *)(malloc (sizeof(unsigned)*n));
    unsigned i2 = 0, i3 = 0, i5 = 0;
    unsigned i;
    unsigned next_multiple_of_2 = 2;
    unsigned next_multiple_of_3 = 3;
    unsigned next_multiple_of_5 = 5;
    unsigned next_ugly_no = 1;
    *(ugly+0) = 1;

    for(i=1; i<n; i++)
    {
       next_ugly_no = min(next_multiple_of_2,
                           next_multiple_of_3,
                           next_multiple_of_5);
       *(ugly+i) = next_ugly_no;
       if(next_ugly_no == next_multiple_of_2)
       {
           i2 = i2+1;
           next_multiple_of_2 = *(ugly+i2)*2;
       }
       if(next_ugly_no == next_multiple_of_3)
       {
           i3 = i3+1;
           next_multiple_of_3 = *(ugly+i3)*3;
       }
       if(next_ugly_no == next_multiple_of_5)
       {
           i5 = i5+1;
           next_multiple_of_5 = *(ugly+i5)*5;
       }
    } /*End of for loop (i=1; i<n; i++) */
    return next_ugly_no;
}

/* Function to find minimum of 3 numbers */
unsigned min(unsigned a, unsigned b, unsigned c)
{
    if(a <= b)
    {
      if(a <= c)
        return a;
      else
        return c;
    }
    if(b <= c)
      return b;
    else
      return c;
}

/* Driver program to test above functions */
int main()
```

```
{
    unsigned no = getNthUglyNo(150);
    printf("%dth ugly no. is %d ", 150, no);
    getchar();
    return 0;
}
```

Algorithmic Paradigm: Dynamic Programming
Time Complexity: O(n)
Storage Complexity: O(n)

Please write comments if you find any bug in the above program or other ways to solve the same problem.

Related Topics:

- Linearity of Expectation
- Iterative Tower of Hanoi
- Count possible ways to construct buildings
- Build Lowest Number by Removing n digits from a given number
- Set Cover Problem | Set 1 (Greedy Approximate Algorithm)
- Find number of days between two given dates
- How to print maximum number of A's using given four keys
- Write an iterative O(Log y) function for pow(x, y)

Tags: Dynamic Programming

Tweet 1 g+1 1

Writing code in comment? Please use **ideone.com** and share the link here.

102 Comments GeeksforGeeks 1 Login

Recommend Share Sort by Newest

Join the discussion…

Nikunj Mundhra • 20 days ago

can anyone explain ?
gretaest divisible power of 3 is 3 , where as for 2 it is 4 and for 5 it is 25.

• Reply • Share ›

nUMEROuNO Nikunj Mundhra • 15 days ago

I had the same doubt. Then I figured that this statement is actually specific to no. 300. You can't divide 300 by 8 or 9 or 125. So, greatest powers of 2, 3 and 5 that divide 300 are 4, 3 and 25 respectively.

• Reply • Share ›

Reply Share ›

Ankur • 21 days ago

@GeeksforGeeks method 2 considers 14 as an ugly number, which is not. could please confirm from your side and suggest alternative algorithm to handle such scenarios?

• Reply • Share ›

Eknoor Ankur • 20 days ago

It does not actually. I had the same same doubt. The method is not multiplying 7*2 after 6*2. i.e. The numbers which are being multiplied with 2,3,and 5 are those numbers which already have been generated in the past series. So after 6*2. the next numbers are 8*2, 5*3 and 3*5 becuase 8,5,3 are already generated and not 7. So the next number is 15 not 14.

• Reply • Share ›

munjal • a month ago

I don't see how the last algo takes O(n) extra space.

• Reply • Share ›

Mohit Verma • a month ago

Try this... now this is printing till 15... but u can print any no. of ugly..

```
public static void printUglyNumsOpt(){

int lastUgly=1;

int multi_2 = 2, multi_3=3, multi_5=5;

int i2 =1, i3=1, i5=1;

int count = 1 , i = 2;

System.out.println("Ugly Numbers:\n");

System.out.println(1);

while(count < 15){

if(multi_2 < multi_3){

if(multi_2 < multi_5){
```

see more

• Reply • Share ›

AJ • 2 months ago

What would be the time complexity be like for this code?

http://ideone.com/2HbYBs

^ | ⌄ • Reply • Share ›

Vishal Chaudhary • 2 months ago

```
#include <iostream>
#include <cstdlib>
#include <cstring>
bool u(int i, int a[100000]){
if(a[i] == 1) return 1;
else if(a[i] == 0) return 0;
else{
if(i%2 != 0 && i%3 != 0 && i%5 != 0){
a[i] = 0;
return 0;
}
if(i == 2 || i == 3 || i == 5){
a[i] = 1;
return 1;
}
int l = 0, m = 0, n = 0;
if(i > 2 && i % 2 == 0) l = u(i/2, a);
if(i > 3 && i % 3 == 0) m = u(i/3, a);
```

see more

^ | ⌄ • Reply • Share ›

shubham gupta • 4 months ago

@admin: in algorithm part, ugly[i] = next_ugly_no shuld be just after computing next ugly number.

2 ^ | ⌄ • Reply • Share ›

Eknoor ➦ shubham gupta • 3 months ago

Doesnot matter as long as its before the next iteration of the loop :)

^ | ⌄ • Reply • Share ›

Abhinav Rastogi • 4 months ago

Why is 14 not an ugly number?

1 ^ | ⌄ • Reply • Share ›

Xueyang Liu ➦ Abhinav Rastogi • 4 months ago

because 14 == 2 * 7; while ugly number's prime factors should only be 2 3 and 5.

1 ^ | ⌄ • Reply • Share ›

AbhishekPathak • 4 months ago

Abhishek Pathak • 4 months ago

```
int ugly_number(int n) {

// only multiples of 2,3,5
// run time = o(n)
// space = o(n)

int[] sub = new int[n];

int ind_2 = 1;

int ind_3 = 0;

int ind_5 = 0;

sub[0] = 1;

sub[1] = 2;

sub[2] = 3;
```

see more

• Reply • Share ›

Ajay Gaur • 4 months ago

I haven't read the algo here, but isn't this can be done using sieve of Eratosthenes (little modified.)?

1) Take an boolean array of length 'n'. (all false initially)
2) initialize array[1] = true
3) array[1*2] = true; array[1*3] = true; array[1*5] = true;
4) then iterate till the number we want to generate ugli numbers, whenever we encounter 'true' anywhere in the array, then take that index and initialize array[2*index] = true; array[3*index] = true; array[5*index] = true.
5) At last, start from 1 and count the 'true' in the arrays, whenever count reaches 150, return the 'index'.

Please correct me if I am wrong.

• Reply • Share ›

Guest ➦ Ajay Gaur • 4 months ago

This will take hell amount of space and will become very inefficient ..
here's how
suppose you want to find n = 30
some where you found true at index =5 you'll then mark 10 (for 2)
10 will mark 20 and 20 will mark 40 (out of size now)

you have to keep doing it till your count become 30 which will exceed size exponentially

^ | ˅ • Reply • Share ›

Amit Kehri • 5 months ago

This is not correct. The sequence is creating 14 also .. 5*2, 6*2, 7*2...
But 14 is not an ugly number. Please correct this algo.

3 ^ | ˅ • Reply • Share ›

Himanshu Mantri → Amit Kehri • 3 months ago

```
if (next_ugly_no == next_mulitple_of_2)
{
i2 = i2 + 1;
next_mulitple_of_2 = ugly[i2]*2;
}
if (next_ugly_no == next_mulitple_of_3)
{
i3 = i3 + 1;
next_mulitple_of_3 = ugly[i3]*3;
}
if(next_ugly_no == next_mulitple_of_5)
{
i5 = i5 + 1;
next_mulitple_of_5 = ugly[i5]*5;
}
```

You are using 'else if' instead of 'if' in the above code block probably. I did the same mistake. :-D

1 ^ | ˅ • Reply • Share ›

JBourne → Himanshu Mantri • 21 days ago

even if you use 'if' you get 14 as one of the ugly number.

^ | ˅ • Reply • Share ›

Priyal Rathi • 5 months ago

Another approach:

We can keep 3 queues Q2 (initially contains 2) , Q3(initially contains 3), Q5(initially contains 5) each time dequeue x = mininum of first element from each queue.

if(x is dequeued from q2) enqueue 2*x in q2, 3*x in q3 and 5*x in q5
if(x is dequeued from q3) enqueue 3*x in q3 and 5*x in q5
if(x is dequeud from q5) enqueue 5*x in q5
and so on.

nth element poped = nth ugly number.

Time complexity: O(n)
Space complexity: O(n)

Link of code: http://ideone.com/euVVFo

2 ⌃ | ⌄ • Reply • Share ›

Amit Kehri ➦ Priyal Rathi • 5 months ago

Thats also wrong u idiot. 3*x in q will result in 3,9,27 ..and so on. Now where the hell is 12,15,18 ?

Actually it should be 3+x, 5+x etc .. now 3*x,5*x.

⌃ | ⌄ • Reply • Share ›

Priya ➦ Amit Kehri • 3 months ago

Amit the above approach is correct.Take pen paper and then try to understand.It is covering 12,15,18 mentioned by you very well.

⌃ | ⌄ • Reply • Share ›

arvind kumar • 6 months ago

This one is also working solution :
http://ideone.com/HIZWpw

⌃ | ⌄ • Reply • Share ›

Ekta Goel • 7 months ago

Even this is gonna work : http://ideone.com/ojbpYr

⌃ | ⌄ • Reply • Share ›

Saurabh Anand ➦ Ekta Goel • 6 months ago

The 150th ugly number is asked, not all the ugly numbers less than 150.

⌃ | ⌄ • Reply • Share ›

Ekta Goel ➦ Saurabh Anand • 6 months ago

Even that can be done easily with little modification.

⌃ | ⌄ • Reply • Share ›

pritika • 7 months ago

I made an algorithm

Just want to check if it is fine in terms of time

public class UglyNum{

public static void main(String args[]){

```
Check c = new Check();

c.get(11);

}

}

class Check{

int []mem = new int[1000];

void initialize(){
```

see more

^ | ⌄ • Reply • Share ›

Amit Kehri ➦ pritika • 5 months ago

WTF .. R u writing Ramayana here ?? First learn to write a neat and clean algorithm .. then come here .

^ | ⌄ • Reply • Share ›

Amit Kehri ka baap ➦ Amit Kehri • 4 months ago

Amit Kehri .. beta tameez nahi sikhaai tumhey tumhaari maa ne ?

1 ^ | ⌄ • Reply • Share ›

aa1992 • 7 months ago

6 is generated twice in the algorithm.

2 ^ | ⌄ • Reply • Share ›

Kim Jong-il • 8 months ago

@GeeksforGeeks
There is a mistake in pseudo code of dynamic programming.
In the step 5: (ugly[i] = next_ugly_no) this statement should be in beginning. not in the last.

3 ^ | ⌄ • Reply • Share ›

Caffrey ➦ Kim Jong-il • 2 months ago

True. It should be beneath
next_ugly_no = Min(next_mulitple_of_2,
next_mulitple_of_3,
next_mulitple_of_5);

^ | ⌄ • Reply • Share ›

don • 8 months ago

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

```
int fun(int num)

{

int n=num;

while(n%2==0)

n/=2;

while(n%3==0)

n/=3;

while(n%5==0)

n=n/5;

if(n<6)
```

see more

• Reply • Share ›

sukanya • 8 months ago

http://ideone.com/Vj52TB

• Reply • Share ›

Shrinivas • 9 months ago

Method 1 can be improved further:
We can reduce number even faster.

```
/* Function to check if a number is ugly or not */
int isUgly(int no){
no = maxDivide(no, 2*3*5);
no = maxDivide(no, 3*5);
no = maxDivide(no, 2*5);
no = maxDivide(no, 2*3);
no = maxDivide(no, 5)
no = maxDivide(no, 3);
no = maxDivide(no, 2);
return (no == 1)? 1 : 0;
}
```

• Reply • Share ›

spidey • 9 months ago

Method2: the code is correct but there is a mistake in the algorithm.

"ugly[i] = next_ugly_no "

This statement should be the second statement in the for loop. Please change it, people may get confused !
:)

2 ⌃ | ⌄ • Reply • Share ›

nalla ↷ spidey • 8 months ago

Yes there is some confusion regarding this in the code

1 ⌃ | ⌄ • Reply • Share ›

Shrinivas ↷ spidey • 9 months ago

Since, variable next_ugly_no gets value assigned only once(at beginning of loop), we can put "ugly[i] = next_ugly_no " any where in loop.

⌃ | ⌄ • Reply • Share ›

Satya ↷ Shrinivas • 9 months ago

I agree with Spidey.

```
if (next_ugly_no == next_mulitple_of_2)
{
i2 = i2 + 1;
next_mulitple_of_2 = ugly[i2]*2;
}
```

If we don't story next_ugly_no (4) into ugly[1],
next_multiple_of_2 would be garbage as ugly[1] won't be initialized.

Code is correct.

1 ⌃ | ⌄ • Reply • Share ›

Guest • 9 months ago

How is 1 an ugly no. ?

⌃ | ⌄ • Reply • Share ›

bon ↷ Guest • 9 months ago

2^0

⌃ | ⌄ • Reply • Share ›

Gaurav Jain • 9 months ago

A much crispier solution using a set (as a min-heap). Although, it does use more memory in absolute terms, asymptotic space complexity is still O(n).

https://github.com/jaingaurav1...

⌃ | ⌄ • Reply • Share ›

Abhijit Sinha ↷ Gaurav Jain • 9 months ago

Abhijit Sinha ↪ Gaurav Jain • 9 months ago

May be I did not understand the question completely: but to me the problem looks much simpler: "like merge sort". Since printing of all ugly numbers is not required hence storing is not useful.

Below Pseudocode no different than above solved solution 2, but uses space O(3)

```
struct Node{
int val;
int mulNo;
}

Node generate_mul_2(){
static int i = 1;

Node n = new Node();
n.val = i++ * 2;
n.mulNo = 2;
return n;
}
```

see more

• Reply • Share ›

shrikanr ↪ Gaurav Jain • 9 months ago

ur code considered a number more than once as ugly number.
e.g. 6 is considered an ugly number twice..

• Reply • Share ›

Gaurav Jain ↪ shrikanr • 9 months ago

It's a set, so it holds only unique elements. However, you are right that '6' is generated more than once, that's why I used a 'set' in the first place.

• Reply • Share ›

Vãîbhåv Joshî • 10 months ago

DP solution in javaa

http://ideone.com/80a7cL

1 • Reply • Share ›

Jun • 10 months ago

Method 2...thoda simple...away from pointers

http://ideone.com/ky7YL5

1 • Reply • Share ›

Jun • 10 months ago

Thoda easy method 1

http://ideone.com/kwYGl2

1 ^ | ˅ • Reply • Share ›

Deepesh Panjabi • a year ago

http://ideone.com/imVOV3

^ | ˅ • Reply • Share ›

ishan • a year ago

In the dp algorithm, the space complexity is not O(n) but O(2^lg(n)). While the two might seem the same but complexity is always written in terms of input size. Here size of input is lg(n), not 'n'.

1 ^ | ˅ • Reply • Share ›

Anindya Dutta ➦ ishan • 3 months ago

2^lg(n) = n. Since lg to the base 2.

^ | ˅ • Reply • Share ›

Load more comments

✉ Subscribe   Add Disqus to your site   Privacy

•

- Interview Experiences
- Advanced Data Structures
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Mathematical Algorithms
- Recursion
- Geometric Algorithms

Popular Posts

- All permutations of a given string
- Memory Layout of C Programs
- Understanding “extern” keyword in C
- Median of two sorted arrays
- Tree traversal without recursion and without stack!
- Structure Member Alignment, Padding and Data Packing
- Intersection point of two Linked Lists
- Lowest Common Ancestor in a BST.
- Check if a binary tree is BST or not
- Sorted Linked List to Balanced BST

Follow @GeeksforGeeks

Recent Comments

- lt_k

  i need help for coding this function in java...

  Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago

- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

- lucy

because we first fill zero in first col and...

Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

- lucy

  @GeeksforGeeks i don't n know what is this long...

  Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 3 hours ago

- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 3 hours ago

-