

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 150th 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.

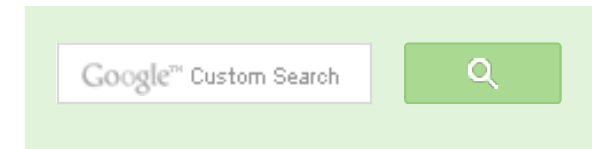
To check if a number is ugly, divide the number by greatest divisible powers of 2, 3 and 5, if the number becomes 1 then it is an ugly number otherwise not.

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>

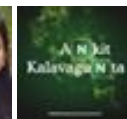
/*This function divides a by greatest divisible
power of b*/
```



GeeksforGeeks



53,523 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)



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

```
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;
}
```

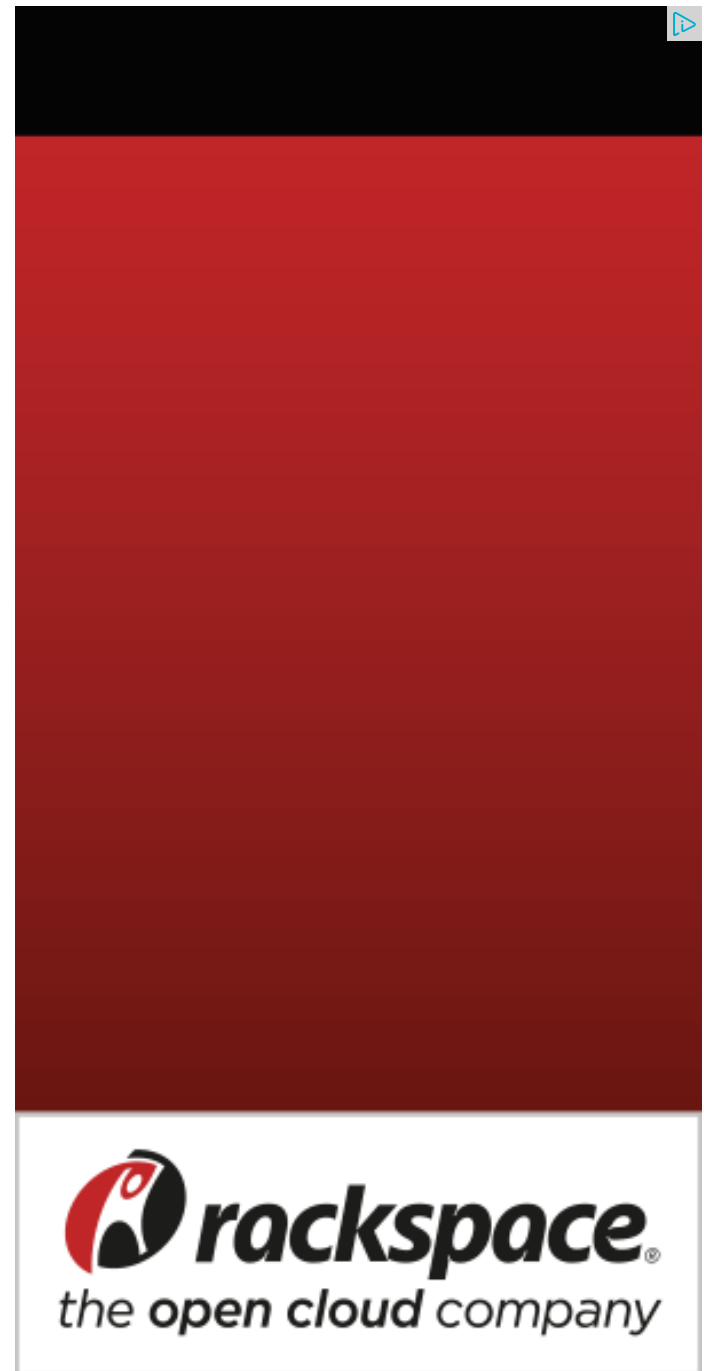
This method is not time efficient as it checks for all integers until ugly number count becomes n, but space complexity of this method is $O(1)$

METHOD 2 (Use Dynamic Programming)

Here is a time efficient solution with $O(n)$ extra space

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_multiple_of_2 = ugly[i2]*2;
    next_multiple_of_3 = ugly[i3]*3;
    next_multiple_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_multiple_of_2,
                       next_multiple_of_3,
                       next_multiple_of_5);
    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)
    {
```





Recent Comments

Aman Hi, Why arent we checking for conditions...

[Write a C program to Delete a Tree.](#) · 37 minutes ago

kzs please provide solution for the problem...

[Backtracking | Set 2 \(Rat in a Maze\)](#) · 41 minutes ago

Sanjay Agarwal bool

[tree::Root_to_leaf_path_given_sum\(tree...](#)
[Root to leaf path sum equal to a given number](#) · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

[Count trailing zeroes in factorial of a number](#) · 1 hour ago

newCoder3006 If the array contains negative numbers also. We...

[Find subarray with given sum](#) · 1 hour ago

newCoder3006 Code without using while loop. We can do it..

[Find subarray with given sum](#) · 1 hour ago

```

        i5 = i5 + 1;
        next_multitple_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;


```

AdChoices 

[► Prime Numbers](#)

[► Generate Code](#)


[► Java Source Code](#)

AdChoices 

[► Java Run Time](#)

[► Code 3](#)

[► Numbers To](#)

AdChoices 

[► VHDL Code](#)

[► Numbers Number](#)

[► C Code](#)

```

*(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;
}

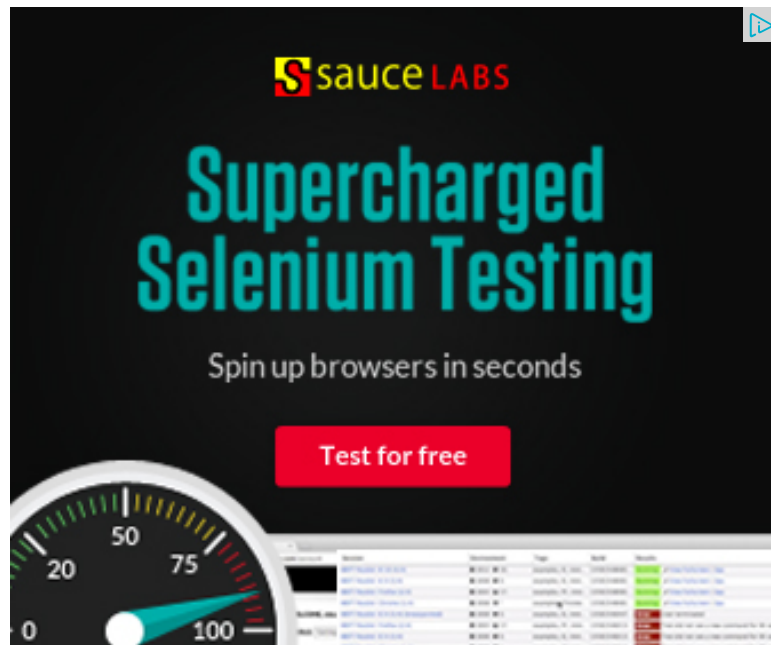
```

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:

- [Backtracking | Set 8 \(Solving Cryptarithmic Puzzles\)](#)
- [Tail Recursion](#)
- [Find if two rectangles overlap](#)
- [Analysis of Algorithm | Set 4 \(Solving Recurrences\)](#)
- [Print all possible paths from top left to bottom right of a mXn matrix](#)
- [Generate all unique partitions of an integer](#)
- [Russian Peasant Multiplication](#)

- [Closest Pair of Points | O\(nlogn\) Implementation](#)



10



Tweet

1



1

Writing code in comment? Please use [ideone.com](#) and share the link here.

@geeksforgeeks, **Some rights reserved**

Contact Us!

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team