CRAZINESS OVERLOADED

Problem statement :

Ziya wants to findout few crazy numbers .A crazy number is a positive integer whose prime factors are restricted to 2, 3, and 5. Ziya wants to perform this task for **T** times Each time an integer **N** is given by ziya ,return the Nth crazy number.
Note:Assume that the sequence generated must be strictly INCREASING order.
Example1:

```
If N = 10

Output: 12

Explanation: [1, 2, 3, 4, 5, 6, 8, 9, 10, 12] is the sequence of the first 10 crazy numbers.



Input :N=1

Output: 1

Explanation: 1 has no prime factors, therefore all of its prime factors are restricted to 2, 3,
and 5.
```

Input Format

- The first line of input will contain a single integer ***T***, denoting the number of test cases.
- Each test case consists of a single line of input containing integer **N **

Constraints

- **1 <= T <=1000**
- **1 <= N <= 1690**

Output Format

- For each test case, output the nth crazy number

Sample Input 0

```
2101
```

Sample Output 0

```
121
```

Explanation 0

Explanation: [1, 2, 3, 4, 5, 6, 8, 9, 10, 12] is the sequence of the first 10 crazy numbers.

```
1 has no prime factors so it is included

2 has one prime factors 2 which is restricted to 2,3,5

3 has one prime factors 3 which is restricted to 2,3,5

4 has one prime factors 2 which is restricted to 2,3,5

5 has two prime factors 2,5 which is restricted to 2,3,5

6 has two prime factors 2,3 which is restricted to 2,3,5

7 has one  prime factors 7 which is not restricted to 2,3,5

8 has one prime factor 2 which is restricted to 2,3,5

9 has one prime factors 3 which is restricted to 2,3,5

10 has two prime factors 2,5 which is restricted to 2,3,5

11 has one prime factor 11 which is not restricted to 2,3,5

12 has two prime factors 2,3 which is restricted to 2,3,5
```

Here is a time efficient solution with O(n) extra space.
The crazy-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:
(1st) 1×2, 2×2, 3×2, 4×2, 5×2, …
(2nd) 1×3, 2×3, 3×3, 4×3, 5×3, …
(3rd) 1×5, 2×5, 3×5, 4×5, 5×5, …
We can find that every subsequence is the crazy-sequence itself (1, 2, 3, 4, 5, …) multiply 2, 3, 5.
Then we use similar merging method to get every crazy number from the three subsequences. Every step we choose the smallest one, and move one step after.

```
1 Declare an array for crazy numbers:  nums[n]
2 Initialize first crazy no:  nums[0] = 1
3 Initialize three array index variables index2, index3, index4 to point to
   1st element of the crazy array:
        index2 = index3 = index5 =0;
4 Initialize 3 choices for the next crazy no:
         next_multiple_of_2 = nums[index2]*2
         next_multiple_of_3 = nums[index3]*3
         next_multiple_of_5 = nums[index5]*5

         Assigning nums[i]=min(all the above three choices);

          i starts from 1 to n-1

          2 4 6 8 10.....

          3 6 9 12 15....

          5 10 15 20 25....

          The bold indicates that 3 choices of multiples and we choose 2 among
them and assign it as nums[1]=2

checking which multiple has matched and increase that index
```

if nums[1] == nums[index2] * 2

  index2++;

if nums[1] == nums[index3] * 3

 index3++;

if nums[1] == nums[index5] * 5

 index5++;

```
here it would be 2==nums[0]*2

hence index2++  {index2=1}
```

now for nums[2]=

   the choices of multiples would be like this

    2 **4** 6 8 10.....

    3 6 9 12 15....

    5 10 15 20 25....

   nums[2]=minimum(nums[index2] * 2,nums[index3] * 3, nums[index5] * 5);

   nums[2]=minimum(nums[1]*2,nums[0]*3,nums[0]*5)

   nums[2]=minimum(2*2,1*3,1*5)

   nums[2]=3;

   and it matches with the nums[i]==nums[index3]*3

   index3++

index3 would be 1

Similarly next pattern will be like

    2 **4** 6 8 10.....

    3 **6** 9 12 15....

    5 10 15 20 25....

nums[3]=4

index2++

next for i=4

    2 4 **6** 8 10.....

    3 **6** 9 12 15....

    5 10 15 20 25....

    nums[4]=5

    index5++

next for i=5;

    2 4 **6** 8 10.....

```
    3 **6** 9 12 15....

    5 **10** 15 20 25....

    nums[5]=6

   here index2 and index3 increases

for i=6

    2 4 6 **8** 10.....

    3 6 **9** 12 15....

    5 **10** 15 20 25....

    nums[6]=8

    index2++
```

Following the same procedure we can generate the sequence of given number and nth number would be present at n-1 th index of the array==>nums[n-1]

HERE GOES THE CODE IN C

```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
int min(int a,int b){

  if(a<b){
    return a;
  }
  else{
    return b;
  }
}
int crazyNumber(int n) {
    int nums[n];
    int index2 = 0, index3 = 0, index5 = 0;
    nums[0] = 1;
    for(int i = 1; i < n; i++){
      nums[i] = min(nums[index2] * 2,min(nums[index3] * 3, nums[index5] * 5));
      if(nums[i] == nums[index2] * 2)
        index2++;
      if(nums[i] == nums[index3] * 3)
        index3++;
      if(nums[i] == nums[index5] * 5)
        index5++;
    }
    return nums[n - 1];
  }
```

```
int main() {
     int t;
      scanf("%d",&t);
         for(int i=0;i<t;i++){
            int n;
             scanf("%d",&n);
            int x=crazyNumber(n);
            printf("%d\n",x);
        }
    return 0;
}
```