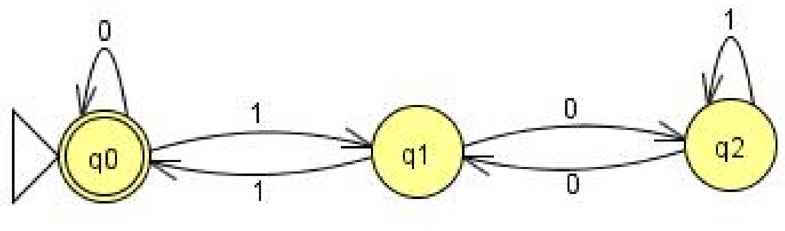
GeeksforGeeks
A computer science portal for geeks

# DFA based division

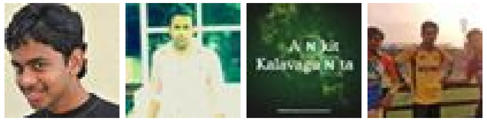
Deterministic Finite Automaton (DFA) can be used to check whether a number “num” is divisible by “k” or not. If the number is not divisible, remainder can also be obtained using DFA.

We consider the binary representation of ‘num’ and build a DFA with k states. The DFA has transition function for both 0 and 1. Once the DFA is built, we process ‘num’ over the DFA to get remainder.

Let us walk through an example. Suppose we want to check whether a given number ‘num’ is divisible by 3 or not. Any number can be written in the form: num = 3*a + b where ‘a’ is the quotient and ‘b’ is the remainder.
For 3, there can be 3 states in DFA, each corresponding to remainder 0, 1 and 2. And each state can have two transitions corresponding 0 and 1 (considering the binary representation of given ‘num’).
The transition function F(p, x) = q tells that on reading alphabet x, we move from state p to state q. Let us name the states as 0, 1 and 2. The initial state will always be 0. The final state indicates the remainder. If the final state is 0, the number is divisible.



In the above diagram, double circled state is final state.



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

1. When we are at state 0 and read 0, we remain at state 0.
2. When we are at state 0 and read 1, we move to state 1, why? The number so formed(1) in decimal gives remainder 1.
3. When we are at state 1 and read 0, we move to state 2, why? The number so formed(10) in decimal gives remainder 2.
4. When we are at state 1 and read 1, we move to state 0, why? The number so formed(11) in decimal gives remainder 0.
5. When we are at state 2 and read 0, we move to state 1, why? The number so formed(100) in decimal gives remainder 1.
6. When we are at state 2 and read 1, we remain at state 2, why? The number so formed(101) in decimal gves remainder 2.

The transition table looks like following:

```
state   0   1
_____________
 0      0   1
 1      2   0
 2      1   2
```

Let us check whether 6 is divisible by 3?
Binary representation of 6 is 110
state = 0
1. state=0, we read 1, new state=1
2. state=1, we read 1, new state=0
3. state=0, we read 0, new state=0
Since the final state is 0, the number is divisible by 3.

Let us take another example number as 4
state=0
1. state=0, we read 1, new state=1
2. state=1, we read 0, new state=2
3. state=2, we read 0, new state=1
Since, the final state is not 0, the number is not divisible by 3. The remainder is 1.

*Note that the final state gives the remainder.*

## 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

We can extend the above solution for any value of k. For a value k, the states would be 0, 1, …. , k-1. How to calculate the transition if the decimal equivalent of the binary bits seen so far, crosses the range k? If we are at state p, we have read p (in decimal). Now we read 0, new read number becomes 2*p. If we read 1, new read number becomes 2*p+1. The new state can be obtained by subtracting k from these values (2p or 2p+1) where 0 <= p < k.
Based on the above approach, following is the working code:

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

// Function to build DFA for divisor k
void preprocess(int k, int Table[][2])
{
    int trans0, trans1;

    // The following loop calculates the two transitions for each stat
    // starting from state 0
    for (int state=0; state<k; ++state)
    {
        // Calculate next state for bit 0
        trans0 = state<<1;
        Table[state][0] = (trans0 < k)? trans0: trans0-k;

        // Calculate next state for bit 1
        trans1 = (state<<1) + 1;
        Table[state][1] = (trans1 < k)? trans1: trans1-k;
    }
}

// A recursive utility function that takes a ‘num’ and DFA (transition
// table) as input and process ‘num’ bit by bit over DFA
void isDivisibleUtil(int num, int* state, int Table[][2])
{
    // process "num" bit by bit from MSB to LSB
    if (num != 0)
    {
        isDivisibleUtil(num>>1, state, Table);
        *state = Table[*state][num&1];
    }
}

// The main function that divides ‘num’ by k and returns the remainder
int isDivisible (int num, int k)
{
    // Allocate memory for transition table. The table will have k*2 e
    int (*Table)[2] = (int (*)[2])malloc(k*sizeof(*Table));

    // Fill the transition table
    preprocess(k, Table);

    // Process ‘num’ over DFA and get the remainder
    int state = 0;
    isDivisibleUtil(num, &state, Table);

    // Note that the final value of state is the remainder
    return state;
}
```

```
// Driver program to test above functions
int main()
{
    int num = 47; // Number to be divided
    int k = 5; // Divisor

    int remainder = isDivisible (num, k);

    if (remainder == 0)
        printf("Divisible\n");
    else
        printf("Not Divisible: Remainder is %d\n", remainder);

    return 0;
}
```

Output:

```
Not Divisible: Remainder is 2
```

DFA based division can be useful if we have a binary stream as input and we want to check for divisibility of the decimal value of stream at any time.

This article is compiled by **Aashish Barnwal** and reviewed by GeeksforGeeks team. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

705 Subscribe

## Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 15 minutes ago

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

Write a C program to Delete a Tree. · 55 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 58 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

## Related Tpoics:

- Backtracking | Set 8 (Solving Cryptarithmetic 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

3 Tweet 0 0

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

**4 Comments** GeeksforGeeks

Sort by Newest ▾

Join the discussion…

**Rahul Singh** · 2 months ago

I want the number to be a binary number say 1001 (9) so how to check the div

**Niko** · 3 months ago

Thanks a lot, this was really helpful!

**Jaka Hudoklin** · 8 months ago

You can replace (trans0 < k)? trans0: trans0-k; with modulo operator. Still thar coursera ;)

**Kelsey** · a year ago

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

typedef struct{

        int zero,one;

}node;

main(){

        int i,divisor;
        int binaryShiftZero,binaryShiftOne;
        node *allNodes;
        printf("What is the divisor?: ");
```

see more

^ | ⌄ •

✉ Subscribe   Add Disqus to your site