Design And Analysis Of Algorithms-Assignment 2 Group-2

ADITYA RAJ(IIT2017005)
MOHIT PAHUJA(IIT2017007)

MAYANK MRINAL(IIT2017006)
ARASHPREET SINGH (IIT2017008)

***Abstract*— This document is complete design and analysis of algorithm to divide a 4 digit number by a 2 digit number given both numbers are positive without using division operator.**

I. INTRODUCTION

In this problem we have to divide a positive 4 digit number with a positive 2 digit number without using division function. For this problem ,we have implemented the traditional long division method. We need to take input of first number as string and second number as integer. We have also used a multiply function , which is working without using multiply operator.

This report further contains::
II. Algorithm Design.
III.Algorithm Analysis
IV.Experimental Study.
V.Conclusions.
VI.References.

II. ALGORITHM DESIGN

A. First Algorithm

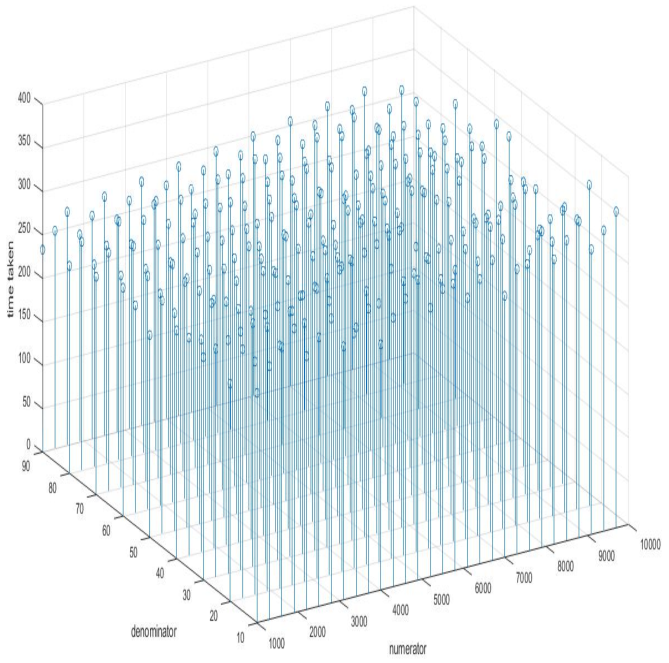
We take input s as string and x as integer.We need to find the remainder and the quotient of s/n.We made an integer z using the first two characters of s.If z is less than x then include the third character of s in z.Search a suitable multiple of s less than or equal to z.Initialize the remainder and quotient and quotient for z and the generated number.Now we get the next digit from s to r till r becomes greater than or equal to x.Generate the new remainder and thus change the quotient.Repeat the process till the last character of s is accessed.The remainder and quotient generated till the end is the final answer.

Multiplication Function : Input : (a,b).
We assign b to c . Now right shift b by 1 bit. Initialize integer ans=0. Add a to ans b times. Double the ans.If c is odd then add a to ans. Therefore ans is our final result.

Algorithm 1

```
  struct{
int v,i}

  multiply(a,b)                                  t=14+5b/2
ans ←0                                           1
c ←b                                             1
b>>=1                                            2
for $i = 1$, $i{+}{+}$, while $i < b$            4+3b/2
ans←ans+a                                        b;

  if c&1 !=0 then
    ans←ans+a
  end if
                                                 total=4
  return ans

  toint(string s,n)                              45n+50
res←0                                            1
for $i = 1$, $i{+}{+}$, while $i < n$            3n+7
res←multiply(res,10) + s[i]-'0'                  42(n+1)
return res

  A search(z,x)                                  74(worst case)
p,i←0                                            2
while 1 do
      temp←p+x                                   2
    if temp≥x then
        break

      else
        p←p+x                                    1
      i←i+1                                      1
      end
  end
                                                 total=70
A ans
ans.v←p                                          1
ans.i←i                                          1
return ans

  main()
n←1                                              1
input←int(x),string(s)
z←toint(s,n)                                     96

  if z≤x then
    z←multiply(z,10) + s[++n]-'0'
  end if
                                                 total=44
A k←search(z,x)                                  75
```

```
r←z-k.v                                  2
q←k.i                                    3
while n<3 do
    r←multiply(r,10) + s[++n] -'0';
    if !s[n] then
        q←multiply(q,10)                  40
    else
        while r<x  n<3 do
            r ←multiply(r,10)+s[++n]-'0'      44
            q←multiply(q,10)                  40
        end
        k ←search(r,x)                    75
        r←r-k.v                           2
        q ←multiply(q,10) + k.i           41
    end
end
print quotient and remainder
                total=constant
```



Thus the graph depicts, our calculation, that is the algorithm takes constant time is correct.

III. ALGORITHM ANALYSIS

A. Time Complexity

- We have calculated the time consumed in each step in terms of number of computations,as given in the above pseudo code.
- The total time calculated was bound by a constant.
- Thus the time complexity in all the cases i.e. best case , average case and worst case is of the order of O(constant) which is equal to O(1).
- Thus our algorithm runs in constant time.

B. Space Complexity

- Since the number of memory location is predefined in our algorithm therefore the space complexity is O(1).

IV. EXPERIMENTAL STUDY

The graph depicts that the units of time taken by our algorithm has a lower bound of approximately 230 and upper bound of approximately 380.

The integrated Development environment of C++ is used for processing the algorithm and graphical analysis is done using MATLAB stem3 function.

V. CONCLUSIONS

In this document we conclude that our algorithm has a constant time complexity, as the lower and the upper bounds are computationally very small.
Therefore both time and space complexity is O(1).

VI. REFERENCES

1.https://en.wikipedia.org/wiki/long division.
2.https://www.theschoolrun.com/what-is-long-division
3.[CLR96] Thomas H. Cormen, Charles E. Leiserson, and Ronald L Rivest. Introduction to algorithms. The MIT press, 2nd edition, 1996.
4.[DW96] Nell Dale and Henry M. Walker. Abstract data types: specifications, implementations, and applications. D. C. Heath and Company, Lexington, MA, USA, 1996.
5.https://stackoverflow.com/questions/33547713/multiplying-two-numbers-without-using-operator-and-without-using-bitwise-in-ti