Alternative Techniques to find Highest Common Factor and Least Common Multiple

R. K. Budhraja[1], Narender Kumar[2]

[1]Department of Mathematics, Sri Venkateswara College, University of Delhi, New Delhi-110021, India
E-mail: *rkbudhraja[at]svc.ac.in*

[2]Department of Mathematics, Aryabhatta College, University of Delhi, Dhaula Kuan, New Delhi-110021, India
E-mail: *nkbudhraja[at]yahoo.com*

Abstract: ***Alternative techniques to find HCF and LCM for a list of numbers are introduced. The approaches are step by step procedures in the form of algorithms, thereby resulting in first ever Computer Program to find HCF and LCM for a given list of numbers. Computer program isformed and executed successfully in C++ language.***

Keywords: HCF, LCM, Prime Factorization, Algorithm

1. Introduction

The HCF of two or more numbers is the largest common factor among all their common factors.

The HCF of two or more given numbers can be obtained by the following methods:
1) By finding first all the factors, then common factors and finally the largest/highest common factor.
2) Prime factorization method.
3) Division Method (based on Euclidean algorithm)[1,2].

The LCM of two or more numbers is the smallest common multiple among all their common multiples. The LCM of two or more given numbers can be obtained by the following methods:
1) By finding first multiples, then common multiples and finally the smallest/least common multiples.
2) Prime Factorization Method.
3) Division Method [1, 2].

The study presents new methods to find HCF and LCM in a different and efficientway. The methods enable way for computer program. Computer program is written in C++ to compute HCF and LCM for a list of given numbers and have successfully been executed to find the same.

2. HCF: Alternate Technique

Method / Algorithm
Step 1. Take any number, preferably the smallest or prime number, among
the given numbers.
(The best number to choose is the one having minimum number of
factors.)
Step 2. List all the factors of this number and arrange these in descending
order.
Step 3. Pick up the largest/ highest factor (the number itself).
Step 4. Check whether this factor is a factor of all the other numbers
(i.e. whether it divides all the other numbers or not?). If yes, then
this factor is the HCF of the given numbers otherwise go to Step 5.
Step 5. Move to the next factor in the list of Step 2 and go to Step 4.
In view of above algorithm, one can redefine the HCF as follows:

Highest Common Factor (HCF)
The highest factor of any number (among given numbers) which is a factor of all the other numbers is the HCF of the given numbers.

The following theorem justifies the algorithm as well as the definition.

Theorem: The number obtained by above algorithm is the HCF.

Proof: Let $N_1, N_2, \ldots, N_m$ be a list of given numbers for which HCF is required.
Then HCF $\{N_1, N_2, \ldots, N_m\} \leq$ Min $\{N_1, N_2, \ldots, N_m\}$
i.e. HCF $\leq N_j \;\forall$ j = 1,2,…,m.

Let N_r be any number from this list of numbers and $f_1(=1)$, $f_2, \ldots, f_p(=N_r)$ be the factors of N_r. The factors in descending order (say)
Factors of N_r: $f_p(=N_r), f_{p-1}, \ldots, f_2, f_1(=1)$
Suppose that f_k ($p \geq k \geq 1$) is the highest factor of N_r which divides all the other numbers.
We assert that HCF = f_k
Obviously HCF $\leq N_r$

The factor f_k is a factor of N_r and it divides all the other numbers and so is a common factor of the given numbers. Further, none of the factors of N_r which are greater than f_k divide all the other numbers. Therefore, no higher factor (greater than f_k) of N_r can be a common factor of the given numbers. Thus, f_k is the highest common factor of the given numbers.

Illustration1:
Let us find the HCF of 45, 80 and 100.
Smallest number = 45.
Factors of 45: 1, 3, 5, 9, 15, 45.
Factors in descending order: 45, 15, 9, 5, 3, 1.
We start with 45.
Now 45 doesn't divide 80.
Discard it and move to next factor 15.
Again 15 doesn't divide 80.
Discard 15 and move to next factor 9.
Again 9 doesn't divide 80.
Discard 9 and we move to next factor 5.
Now 5 divides 80 and also 5 divides 100.
Hence , HCF = 5.

Illustration 2:
Let us find the HCF of 52, 78, 13 and 27.
Prime number = 13.
Factors of 13: 1, 13.
Factors in descending order: 13, 1.
We start with 13.
Now 13 divides 52 and 78 but 13 doesn't divide 27.
Discard it and move to next factor 1.
Obviously, HCF = 1.

3. LCM: Alternate Technique

Method / Algorithm
Step 1. Take any number, preferably the largest number, among the given numbers.
Step 2. List multiples of this number and arrange these multiples in ascending order.
Step 3. Pick up the lowest multiple (the number itself).
Step 4. Check whether this multiple is a multiple of other numbers (i.e. whether this multiple is divisible by all the other numbers?). If yes, then this number is the LCM otherwise go to Step 5.
Step 5. Move to the next multiple in the list of Step 2 and go to Step 4.
In view of above algorithm, one can redefine the LCM as follows:

Lowest Common Multiple (LCM)
The lowest multiple of any number (among given numbers) which is a multiple of all the other numbers is the LCM of the given numbers.
The justification of the algorithm as well as the definition follows on the lines of the above theorem (for HCF).

Illustration 1:
Let us find the LCM of 24, 32, 12.
Largest number = 32.
Multiples of 32: 32, 64, 96, 128, …
We start with 32.
Now 32 is not a multiple of 12 (in other words 12 doesn't divide 32).
Discard 32 and move to next multiple 64.
Again 64 is not a multiple of 12.
Discard 64 and move to next multiple 96.
Now 96 is a multiple of 12 and as well as a multiple of 24.
Hence, LCM = 96.

Illustration 2:
Let us find the LCM of 6, 18, 27, 12.
Largest number = 27.
Multiples of 27 in ascending order: 27, 54, 81, 108, 135, …
We start with 27.
Now 6 doesn't divide 27.
Discard 27 and move to next multiple 54.
Here 6 and 18 divide 54 but 12 doesn't divide 54.
Discard 54 and move to next multiple 81.
Now 6 doesn't divide 81.
Discard 81 and move to next multiple 108.
Here 6, 18 and 12 (all the other numbers) divide 108.
Hence, LCM = 108.

The computational aspect of these algorithms lead to simple computer program (in any of the high level languages) to evaluate HCF and LCM for a given list of numbers.

4. Computer Program

The following is the Computer programfor these approaches to find HCF and LCM in C++ language and has been executed successfully.

```
#include<iostream>
using namespace std;
int main()
{
   int n,count,count2,c=0;
   int a[100];
   cout<<"How many numbers do you wish to input? ";
   cin>>n;
   for(int i = 0; i< n; i++)
        {
              cin>>a[i];
        }

        for(int i=0; i<n-1; i++)
        {
              for(int j = i+1; j<n ; j++)
              {
                    if(a[i] > a[j])
                    {
                          int temp = a[i];
                          a[i] = a[j];
                          a[j] = temp;
                    }
              }
        }

        cout<<"Numbers in Ascending order: "<<endl;
        for(int i = 0; i< n; i++)
        {
              cout<<a[i]<<endl;
        }
        cout<<"and the largest number is "<<a[n-1]<<endl;

        cout<<"The factors of "<<a[0]<<" are "<<endl;
        for(int i=a[0];i>=1;i--)
        {
              if(a[0]%i==0)
                    {
                    cout<<i<<endl;
                    }
```

```
        }
        for(int i=a[0];i>=1;i--)
        {  count=0;
           if(a[0]%i==0)   //this produces factors
                {
                        for(int k=1;k<n;k++)
                        {
                                if(a[k]%i==0)
                                count++;
                            }
                                if(count==n-1)
                                {
                                 cout<<"HCF is "<<i;
                                break;
                                  }

                        }
        }
        cout<<endl;
        count2=0;
        while(count2!=n-1)
                {
                        c++; count2=0;
                        for(int i=0;i<n-1;i++)
                        {
                                if(a[n-1]*c%a[i]==0)
                                count2++;
                        }

                }
        if(count2==n-1)
                {
                      cout<<"LCM is "<<a[n-1]*c;
                }
         return 0;
}
```

5. Remark

The techniques to evaluate HCF and LCM are different from the known methods which we have been using for so many decades. Further these methods are less time consuming and simpler to the existing methods.

References

[1] Hardy, G. H., & Wright, E. M. (1979)*An Introduction to the Theory of Numbers (Fifth edition)* Oxford University Press, Oxford, ISBN 978-0-19-853171-5.
[2] Long, Calvin T. (1972) Elementary Introduction to Number Theory (2nd ed.), D. C. Heath and Company, Lexington, LCCN 77-171950.