CNS LAB

Batch: B1

Assignment: 7

PRN No: 2020BTECS00006

Name: Samrat Vishwas Jadhav

Title of assignment: Implementation of Euclidean and Extended Euclidean Algorithm.

1. **Aim:**
   Implementation of Euclidean and Extended Euclidean Algorithm.

2. **Theory:**

The Euclidean algorithm is a way to find the greatest common divisor of two positive integers. GCD of two numbers is the largest number that divides both of them. A simple way to find GCD is to factorize both numbers and multiply common prime factors.

```
36  = 2 x 2 x 3 x 3
60  = 2 x 2 x 3 x 5

GCD = Multiplication of common factors
    = 2 x 2 x 3
    = 12
```

Basic Euclidean Algorithm for GCD:
The algorithm is based on the below facts.

- If we subtract a smaller number from a larger one (we reduce a larger number), GCD doesn't change. So if we keep subtracting repeatedly the larger of two, we end up with GCD.
- Now instead of subtraction, if we divide the smaller number, the algorithm stops when we find the remainder 0.

Extended Euclidean algorithm also finds integer coefficients x and y such that: ax + by = gcd(a, b)

Examples:

Input: a = 30, b = 20
Output: gcd = 10, x = 1, y = -1
(Note that 30*1 + 20*(-1) = 10)

Input: a = 35, b = 15
Output: gcd = 5, x = 1, y = -2
(Note that 35*1 + 15*(-2) = 5)

How does Extended Algorithm Work?
As seen above, x and y are results for inputs a and b,

$a.x + b.y = gcd$ —-*(1)*

And x_1 and y_1 are results for inputs b%a and a*(b%a).x_1 + a.y_1 = gcd*

When we put b%a = (b – (⌊b/a⌋).a) in above, we get following. Note that ⌊b/a⌋ is floor(b/a) *(b –* (⌊b/a⌋).a).x_1 + a.y_1 = gcd

Above equation can also be written as below

$b.x_1 + a.(y_1 - (\lfloor b/a \rfloor).x_1) = gcd$ —*(2)*

After comparing coefficients of ‘a’ and ‘b’ in (1) and (2), we get following,
$x = y_1 - \lfloor b/a \rfloor * x_1$
$y = x_1$

How is Extended Algorithm Useful?

The extended Euclidean algorithm is particularly useful when a and b are coprime (or gcd is 1). Since x is the modular multiplicative inverse of “a modulo b”, and y is the modular multiplicative inverse of “b modulo a”. In particular, the computation of the modular multiplicative inverse is an essential step in RSA public-key encryption method.

Code:

```
#include<iostream>
#include<bits/stdc++.h>
using namespace std;

class menu
{
    public :
   long long find_multiplicative_inverse(long long a, long long
b) {
    long long q, r, t1 = 0, t2 = 1, t, main_a = a;
cout<<"\n_______________________________________________\n";
    cout << "|\tQ\t|\tA\t|\tB\t|\tR\t|\tT1\t|\tT2\t|\tT\t|\n";
  cout<<"\n_______________________________________________\n";


    while (b > 0) {
        q = a / b;
        r = a % b;
        t = t1 -  (t2 * q );
        cout << "|\t" << q << "\t|\t" << a << "\t|\t" << b <<
"\t|\t" << r << "\t|\t" << t1 << "\t|\t" << t2 << "\t|\t" << t <<
"\t|\n";
      cout<<"\n_______________________________________________\n"
;

        a = b;
        b = r;
        t1 = t2;
        t2 = t;
    }

    cout << "|\t" << q << "\t|\t" << a << "\t|\t" << b << "\t|\t"
<< r << "\t|\t" << t1 << "\t|\t" << t2 << "\t|\t" << t <<
"\t|\n";
    cout<<"\n_______________________________________________\n";
```

```
    if (t1 < 0) {
        t1 += main_a;
    }
    return t1;
}

    long long find_large_number_gcd(long long a,long long b)
    {
        long long q,r;
            cout<<"\n____________________________________________
___\n";
            cout<<"|\t\tQ\t\t|\t\tA\t\t|\t\tB\t\t|\t\tR\t\t|\n";
          cout<<"\n______________________________________________
_\n";

            while(b>0)
            {
                    q=a/b;
                    r=a%b;
                    cout<<"|\t\t"<<q<<"\t\t|\t\t"<<a<<"\t\t|\t\t"
<<b<<"\t\t|\t\t"<<r<<"\t\t|\n";
                 cout<<"\n_______________________________________
________\n";
                    a=b;
                    b=r;
            }
             cout<<"|\t\t"<<q<<"\t\t|\t\t"<<a<<"\t\t|\t\t"<<b<<"\
t\t|\t\t"<<r<<"\t\t|\n";
    cout<<"\n_______________________________________________\n";

            cout<<endl;

            return a;
    }

};
```

```
int main()
{
    main_menu:
    cout<<"\n____________________________\n";
    cout<<"\n1.Find Multiplicative Inverse (Extended Euclidien
Algo ) \n2.Find GCD Of large numbers(Euclideian Algo ) \n";
    cout<<"____________________________\n";
    cout<<"Enter Choice Code :\t";
    menu object;
    int ch;
    cin>>ch;
    cout<<"\n";
    long long a,b,ans;

    switch(ch)
    {
        case 1 :

            cout<<"\nEnter  A and B ( must be A>B)  :\t";
            cin>>a>>b;
            ans=object.find_multiplicative_inverse(a,b);
            cout<<"Multiplicative Inverse Of  "<<a<<"\tAnd
"<<b<<"\t :\t"<<ans<<endl;
            goto main_menu;

        case 2:

            cout<<"\nEnter  A and B  :\t";
            cin>>a>>b;
             ans=object.find_large_number_gcd(a,b);
            cout<<"\nGCD Of   Of  "<<a<<"\tAnd "<<b<<"\t
:\t"<<ans<<endl;
            goto main_menu;

        default:
            cout<<"Invalid Input !";
            break;
```

```
    }
    return 0;

}
```

Output:

```
1.Find Multiplicative Inverse (Extended Euclidien Algo )
2.Find GCD Of large numbers(Euclideian Algo )

____________________________
Enter Choice Code :     1


Enter  A and B ( must be A>B)  :        161
28


-----------------------------------------------
|       Q       |       A       |       B       |       R       |       T1      |       T2      |       T       |

-----------------------------------------------
|       5       |       161     |       28      |       21      |       0       |       1       |       -5      |

-----------------------------------------------
|       1       |       28      |       21      |       7       |       1       |       -5      |       6       |

-----------------------------------------------
|       3       |       21      |       7       |       0       |       -5      |       6       |       -23     |

-----------------------------------------------
|       3       |       7       |       0       |       0       |       6       |       -23     |       -23     |

-----------------------------------------------
Multiplicative Inverse Of  161  And 28    :      6
```

```
____________________________
Enter Choice Code :     2


Enter  A and B  :       2740
1760

-----------------------------------------------
|               Q               |               A               |               B               |               R               |

-----------------------------------------------
|               1               |               2740            |               1760            |               980             |

-----------------------------------------------
|               1               |               1760            |               980             |               780             |

-----------------------------------------------
|               1               |               980             |               780             |               200             |

-----------------------------------------------
|               3               |               780             |               200             |               180             |

-----------------------------------------------
|               1               |               200             |               180             |               20              |

-----------------------------------------------
|               9               |               180             |               20              |               0               |

-----------------------------------------------
|               9               |               20              |               0               |               0               |

-----------------------------------------------


GCD Of   Of  2740        And 1760          :      20
```