Name : Omkar Gurav
Roll no : 8048

Assignment No: 1

Title: Develop and program in C++ based on number theory such as Chinese remainder.

Objective: To study Chinese remainder theorem.

Theory:

Chinese Remainder Theorem :

According to D. Wells, the problem posed by Sun Tsu (4^{th} century AD) there are certain things whose numbers are not known. Suppose there is some number p which is divided by 2, the remainder is 1, if p is divided by 3, the remainder is 1 and if p is divided by 4,5 and 6, the remainder is 1. But if it is divided by 7, the remainder is zero. Then what is the smallest value of p. Chinese remainder theorem is used to get the solution of such problem. Using this theorem we get the value of p.

Theorem :

There are two relatively prime numbers m and n, which are modulo m and n, the congruence

$$p \equiv a \bmod m$$
$$p \equiv b \bmod n$$

have a unique solution : $p \bmod mn$

Suppose $n_1, n_2, \ldots, n_r$ are the relatively prime integer numbers and $b_1, b_2, \ldots, b_r$ are the remainders for $n_1, n_2, \ldots, n_r$ respectively. Then the system of congruence, $p \equiv b_i \pmod{n_i}$ for $1 \le i \le r$, has a unique solution.

$$N = n_1 * n_2 * \ldots * n_r,$$

Which is given by :

$$p \equiv b_1N_1y_1 + b_2N_2y_2 + \ldots + b_rN_ry_r \pmod N,$$

where $N_i = N/n_l$ and

$$y_i = (N_i)^{-1} \pmod{n_i} \text{ for } 1 \le i \le r,$$

where y_i is the multiplicative inverse of $(N_i) \pmod{n_i}$.

Example:

1. Firstly express the problem as a system of congruence,

$$p \equiv b_i \pmod{n_i}$$

where n_i are relatively prime numbers : n_1, n_2, n_3 and so on.
b_i is the respective remainder for modulo n_i such that b_1 for n_1, b_2 for n_2 and so on.
P is the value of solution.

Here ,

$$p \equiv 0 \pmod 2$$
$$p \equiv 1 \pmod 3$$
$$p \equiv 0 \pmod 5$$
$$p \equiv 6 \pmod 7$$
$$p \equiv 6 \pmod{11}$$

2. Calculate the value of $N = n_1 * n_2 * \ldots * n_r$

$N = 2*3*5*7*11 = 2310$

3. Calculate the value of $N_i = N/n_i$ such that $N_1 = N/n_1$, $N_2 = N/n_2$ and so on.

$N_2 = 2310/2 = 1155$
$N_3 = 2310/3 = 770$
$N_5 = 2310/5 = 462$
$N_7 = 2310/7 = 330$
$N_{11} = 2310/11 = 210$

4. Calculate the multiplicative inverse for $y_i \equiv (N_i)^{-1}(\text{mod } n_i)$
where y_i is the multiplicative inverse of $(N_i)(\text{mod } n_i)$.

$y_2 = (1155)^{-1}(\text{mod } 2) = 1$
$y_3 = (770)^{-1}(\text{mod } 3) = 2$
$y_5 = (462)^{-1}(\text{mod } 5) = 3$
$y_7 = (330)^{-1}(\text{mod } 7) = 1$
$y_{11} = (210)^{-1}(\text{mod } 11) = 1$

5. The value of p is calculated as :

$$p \equiv (b_1N_1y_1 + b_2N_2y_2 + \ldots + b_rN_ry_r) \text{ mod } N$$

where, p is the solution of the problem.

$p \equiv 0(1155)(1) + 1(770)(2) + 0(462)(3) + 6(330)(1) + 6(210)(1)$
$p = 0 + 1540 + 0 + 1980 + 1260$
$p = 4780 \text{ mod } 2310 = 160$

The value of p is 160.

Input:

- Number of divisors N = 3
- Values of divisors and remainders
  $n_1 = 5$, $n_2 = 7$, $n_3 = 8$
  $b_1 = 3$, $b_2 = 1$, $b_3 = 6$

Output:

- Total product of divisors total_product = 280
-

bi	Ni	yi	biNiyi
3	56	1	168
1	40	3	120
6	35	3	630

- Number is 78

Algorithm:

1. Start
2. Input number of divisors and values of divisors and remainders.

3. Check if all pairs of divisors are coprime.
4. Calculate total product of all divisors.
5. Show total product.
6. Calculate N_i for every divisor.
7. Calculate modular multiplicative inverse of each N_i.
8. Calculate multiplication of remainder, N_i and modular multiplicative inverse for each remainder.
9. Calculate addition of all multiplications in step 8.
10. Calculate number using (addition mod total product).
11. Show table of bi Ni yi biNiyi.
12. Show number.
13. Stop.

Conclusion: We have studied and implemented the CRT - Chinese Remainder Theorem.

Chinese_remainder.cpp

```
#include<iostream>

using namespace std;



class chinese_remainder

{

   public:


   int total_product = 1;

   int N;

   int Ni[100];

   int yi[100];

   int n[100];
```

```
    int b[100];
    void get_data();
    void evaluate();
    bool relative_prime();
    static int gcd(int a, int b);

};



void chinese_remainder::get_data()
{

    cout << "\nEnter no. of divisors : ";
    cin >> N;

    cout << "\nEnter values of divisor (n) : ";

    for(int i = 0; i < N; i++)
    {
        cin >> n[i];
    }

    cout<<"\nEnter values of remainder (b) : ";

    for (int i = 0; i < N; i++)
    {
        cin >> b[i];
    }
```

```
}


int chinese_remainder::gcd(int a, int b)
{
   if (b == 0)
   return a;

   return gcd(b, a%b);
}


bool chinese_remainder::relative_prime()
{

   for(int i = 0; i < N - 1; i++)
   {
      for(int j = i + 1; j < N; j++)
      {
         if( gcd(n[i], n[j]) != 1)
           return false;
      }

   }

   return true;
}
```

```
void chinese_remainder::evaluate()
{
    int ans = 0, j =1;
    float k = 2.4;

    // Calculating total product of divisors
    for (int i = 0; i < N; i++)
    {
        total_product = total_product * n[i] ;
    }

    cout << "\nTotal product of divisors (n) is : " << total_product << "\n";

    for (int i = 0; i < N; i++)
    {
        Ni[i] = total_product/n[i];
    }

    cout<<"\n";


    for (int i = 0; i < N; i++)
    {
        yi[i] = 1;
    }
```

```
cout<<"\n";

for (int i = 0; i < N; i++)
{
   // Loop for finding out modular multiplicative inverse

   for(int j = 1; j < n[i]; j++)
   {
      if((Ni[i]*j) % n[i] == 1)
         yi[i] = j;
   }

}

for (int i = 0; i < N; i++)
{
   ans += b[i]*Ni[i]*yi[i] ;
}

ans = ans % total_product;

cout << "\nbi\tNi\tyi\tbiNiyi";

for (int i = 0; i < N; i++)
{
   cout << "\n\n" << b[i] << "\t" << Ni[i] << "\t" << yi[i] << "\t" << b[i]*Ni[i]*yi[i];
}
```

```
    cout<<"\n\nNumber is " << ans;
}



int main()
{
    chinese_remainder c;
    c.get_data();

    if(c.relative_prime())
    {
        c.evaluate();
    }
    else
    {
        cout << "Divisors (n) are not relative prime !!!\n";
    }

    return 0;

}
```

Output :

```
Enter no. of divisors : 3
```

Enter values of divisor (n) : 3 4 5

Enter values of remainder (b) : 2 3 1

Total product of divisors (n) is : 60

bi	Ni	yi	biNiyi
2	20	2	80
3	15	3	135
1	12	3	36

Number is 11