Ayush Goyal

190905522 CSE D 62

DAA LAB 2 (Week 2)

Q1) Write a program to find GCD using consecutive integer checking method and analyze its time efficiency.

CODE:

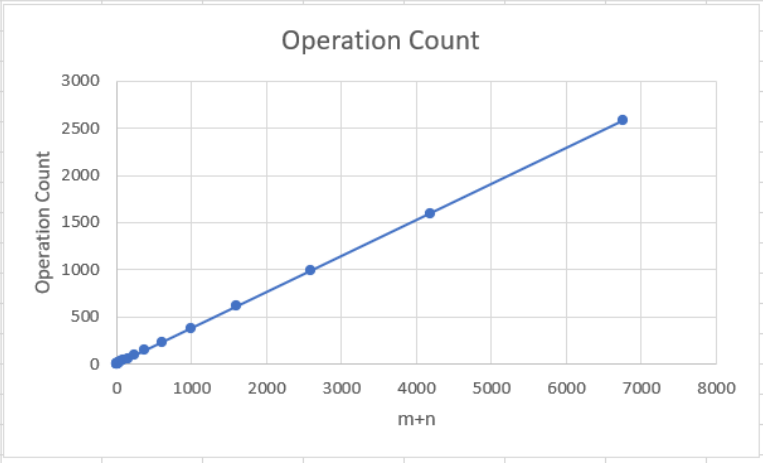
```
#include <stdio.h>
#include <stdlib.h>

int gcd(int a, int b, int *x)
{
  for (int i = (a > b ? b : a); i > 0; i--)
  {
    (*x)++;
    if (a % i == 0 && b % i == 0)
    {
      return i;
    }
  }
}

int main()
{
  int m, n;
  scanf("%d %d", &m, &n);
  int x = 0;
  int result = gcd(m, n, &x);
  printf("\nGCD is %d and the Opcount is  %d\n", result, x);
}
```

OUTPUT:

```
C:\Users\HP\Desktop\CSE\DAA Lab>gcc consecutive.c -o consecutive

C:\Users\HP\Desktop\CSE\DAA Lab>consecutive
1 2

GCD is 1 and the Opcount is  1

C:\Users\HP\Desktop\CSE\DAA Lab>consecutive
89 144

GCD is 1 and the Opcount is  89

C:\Users\HP\Desktop\CSE\DAA Lab>consecutive
4181 6765

GCD is 1 and the Opcount is  4181

C:\Users\HP\Desktop\CSE\DAA Lab>
```

Table and Graph for Time Efficiency ((m+n) vs Operation Count):

m+n	Operation Count
1	1
2	1
3	1
5	2
8	3
13	5
21	8
34	13
55	21
89	34
144	55
233	89
377	144
610	233
987	377
1597	610
2584	987
4181	1597
6765	2584



Q2) Write a program to find GCD using middle school method and analyze its time efficiency.

CODE:

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

void sievesalgo(int m, int arr[])
{
  for (int i = 2; i < m + 1; i++)
  {
    arr[i] = i;
  }
  int j;
  for (int i = 2; i < m + 1; i++)
  {
    if (arr[i] != 0)
    {
      j = i * i;
      while (j <= m)
      {
        arr[j] = 0;
        j = j + i;
      }
    }
  }
}

int primefactors(int n, int arr[], int *op)
{
  int narr[n + 1];
  sievesalgo(n, narr);
  int i = 2;
```

```
  int cnt = 0;
  while (i <= n)
  {
    (*op)++;
    if (narr[i] != 0)
    {
      if (n % narr[i] == 0)
      {
        arr[cnt] = narr[i];
        n = n / narr[i];
        cnt++;
      }
      else
      {
        i++;
      }
    }
    else
    {
      i++;
    }
  }
  return cnt;
}

int gcd(int m, int n, int *opcount)
{
  if (m == 0 || n == 0)
  {
    *opcount = 1;
    return m == 0 ? n : m;
```

```
 }
 int marr[m], narr[n], op1 = 0, op2 = 0;
 int a = primefactors(m, marr, &op1);
 int b = primefactors(n, narr, &op2);
 *opcount = op1 + op2;
 printf("\n");
 int i = 0, j = 0;
 int res = 1;
 while (i < a && j < b)
 {
  if (marr[i] == narr[j])
  {
   res *= marr[i];
   i++;
   j++;
  }
  else if (marr[i] < narr[j])
  {
   i++;
  }
  else
  {
   j++;
  }
 }
 return res;
}

int main()
{
 int x, y;
```

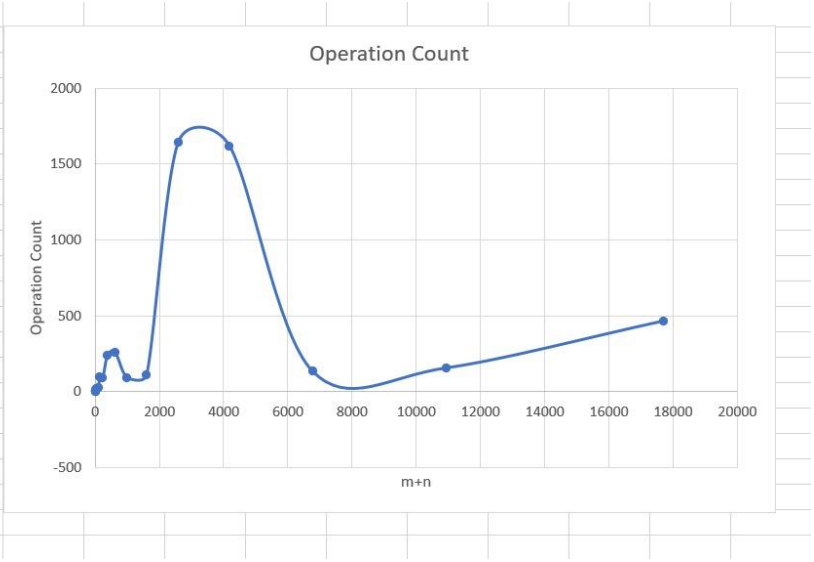
```
  scanf("%d %d", &x, &y);
  int op = 0;
  int res = gcd(x, y, &op);
  printf("GCD is %d and the opcount is : %d\n", res, op);
}
```

OUTPUT:

```
C:\Users\HP\Desktop\CSE\DAA Lab>gcc middleschool.c -o middleschool

C:\Users\HP\Desktop\CSE\DAA Lab>middleschool
3 5

GCD is 1 and the opcount is : 6

C:\Users\HP\Desktop\CSE\DAA Lab>middleschool
55 89

GCD is 1 and the opcount is : 99

C:\Users\HP\Desktop\CSE\DAA Lab>middleschool
377 610

GCD is 1 and the opcount is : 91

C:\Users\HP\Desktop\CSE\DAA Lab>middleschool
4181 6765

GCD is 1 and the opcount is : 156

C:\Users\HP\Desktop\CSE\DAA Lab>middleschool
6765 10946

GCD is 1 and the opcount is : 465

C:\Users\HP\Desktop\CSE\DAA Lab>
```

Table and Graph:

m	n	m+n	Operati
0	1	1	1
1	1	2	0
1	2	3	1
2	3	5	3
3	5	8	6
5	8	13	7
8	13	21	15
13	21	34	19
21	34	55	24
34	55	89	28
55	89	144	99
89	144	233	95
144	233	377	239
233	377	610	261
377	610	987	91
610	987	1597	110
987	1597	2584	1644
1597	2584	4181	1618
2584	4181	6765	135
4181	6765	10946	156
6765	10946	17711	465



THE END