**5. Topic – Multiplications of large Strings using C program**

* **Problem Statement**

Write a program to multiply two integers a and b of length m and n respectively. Take m, n, a and b as input from the user. Use dynamic memory allocation for storing those big integers.

**Input example:** /\* Here user will give the lengths of the strings and the strings of that lengths\*/

Enter the length of the first integer

8

Enter the integer

12346789

Enter the length of the second integer

7

Enter the integer

3456123

**Output example:**

\*\*\*\*\*\*\* The product of the given numbers are : 42672021439047\*\*\*\*\*\*\*\*\*\*

* **Proposed C Code**

**/\* ---------- multiplicationofstrings.c--------------- \*/**

#include<stdio.h>

#include<stdlib.h>

int main()

{

/\* The length of the two integers are taken from user \*/

int n,m;

printf("Enter the length of the first integer\n");

scanf("%d",&m);

/\* getchar() is used to avoid \n as an input to string \*/

getchar();

/\* The string is created dynamically to store the big integer as character array dynamically \*/

char\* a = (char\*)malloc(m+1);

printf("Enter the integer\n");

scanf("%s",a);

/\* In the same way of previous one the second integer is stored in a string \*/

printf("Enter the length of the second integer\n");

scanf("%d",&n);

getchar();

char\* b = (char\*)malloc(n+1);

printf("Enter the integer\n");

scanf("%s",b);

/\* The 2D array result is created dynamically and initialized to zero to perform multiplication of every digit to digit \*/

int\*\* result = (int \*\*)calloc(n\*(m+n),sizeof(int\*));

int i,j;

for ( i =0 ; i < n ; i++ )

{

result[i]=(int\*)calloc(m+n,sizeof(int\*));

}

/\* The variables needed for the multiplication of every digit to digit,here carrycount is used to measure carry at every step\*/

int val\_1st,val\_2nd,carrycount=0,multiply,shifter=1,row,col;

/\* shifter is declared as it will be used to shift the multiplication of first string with the digits from second string consecutively starting from the end of the second string \*/

for ( i =n-1 , row = 0 ; i >= 0 ; i-- , row++ )

{

val\_2nd = b[i] - '0';

for ( j = m-1 , col = m+n-shifter ; j >= 0 ; j-- , col-- )

{

val\_1st = a[j] - '0';

multiply = val\_2nd\*val\_1st + carrycount;

result[row][col] = multiply % 10;

carrycount = multiply/10;

}

if(carrycount > 0)

{

result[row][col] = carrycount;

}

shifter++;

carrycount = 0;

}

/\* The product string is created to store the final result as a string \*/

char \* product = (char \*)calloc(m+n+1,sizeof(char));

for ( col = m+n-1 ; col >= 0 ; col-- )

{

int col\_add = 0;

for ( row = n-1 ; row >= 0 ; row-- )

{

col\_add += result[row][col];

/\* Digits are added column wise in the final result \*/

}

col\_add += carrycount;

/\* The generated carry in some steps of column addition is added with the next column \*/

product[col] = (col\_add % 10) + 48;

carrycount =col\_add/10;

}

if(carrycount > 0)

{

product[col] = carrycount;

}

/\*If the product[0] is 0 i.e no carry then we will print the string from incremented ointer product+1 \*/

if(product[0] == '0')

{

product = product + 1;

}

/\* The final result is printed \*/

printf("\n\n\*\*\*\*\*\*\* The product of the given numbers are : %s\*\*\*\*\*\*\*\*\*\*\n\n",product);

return 0;

}

**/\*------------------------------------------------------------------------------------------------------------------------- \*/**

* **Conclusion**

**The proposed algorithm has a runtime of O(n^2) where n is generally input string size under consideration.**

* **Limitations : If we locally declare any array then its size should be less than 10^5 which we have to consider and also we cannot perform the program below O(n^2) as we have to multiply each character by character from the two strings .**
* **Assumptions: Here we are considering the strings to be positive integers .**