**Structure and Analysis of Algorithm**

**Assignment-1**

**Group-2** (**IEC2021005**-**IEC2021009**)

**Gaurav Sahay Ansh Kothari Sonali Gupta Pragya Pal Ayush Chandra**

(IEC2021005) (IEC2021006) (IEC2021007) (IEC2021008) (IEC2021009)

**Introduction to Programming**

# **Indian Institute of Information Technology, Allahabad**

***Abstract*— Given an unsorted array of n numbers, find the sum of all the perfect numbers and check whether it is Armstrong number or not.**

# **Introduction**

Given an array arr[] of size n, we will find the sum of all the perfect numbers and further we will check whether the sum is an Armstrong number or not.

A **perfect number** is a positive integer that is equal to the sum of all its proper positive divisors excluding the number itself. For e.g.: 6,28, 256, etc. 6 is a perfect number because the proper divisor of 6 are 1,2,3 (excluding 6) and sum of all its positive divisors (1+2+3) is 6.

An **Armstrong number** of order n is a number in which each digit when multiplied by itself n number of times and finally added together, results the same number. For eg: 371 is a three-digit number. Therefore, order=3. now, here each digit is multiplied by itself three times and finally added together and results in our original number. For e.g.:

(3\*3\*3+7\*7\*7+1\*1\*1) = (27+343+1) = 371

# **ALGORITHM DESIGN**

* We initiated the program by declaring “b” as a size of array and specified Boolean perfect=false, this will be useful in calculating whether the number is perfect or not.
* Set max array size as a[100]. Then executing program will ask for first number of elements.
* For loop will be executed, asking user to enter that number of elements as per the choice.
* Sum=0, d=0 are stored in integer type variable. sum will store sum of divisors of array elements and d will store sum of all perfect numbers among those array elements.
* To extract the values from array like a[i], for loop is used and sum=0 is stored by default.
* For checking the perfect numbers, another for loop is used. for (int j = 1; j <= a[i] / 2; j=j+1);

Now if statement is used, if a[i] is divided by j and remainder comes as 0 then it will calculate sum of divisors as sum= sum + j.

* if sum of divisors = a[i] itself, then that sum will be considered as a perfect number. And further d will store that sum as d= d + sum.
* Now if perfect stores True then it will further check for Armstrong Number and if not then it will display that there is no perfect number.
* Sum1 and Sum2 are two copies of d and nod will store the number of digits. First while loop will calculate the number of digits in d and store it in the nod.
* Second while loop is to extract each digit of d in e(base) and calculate and store power raised to the digit in digsum.
* If this digsum becomes equal to d then it will be an Armstrong number otherwise it will be not.

# **ALGORITHM AND ILLUSTRATION**

Begin

int b

Bool Value of perfect=FALSE

Array a[100]

DISPLAY- “Enter number of elements”

INPUT b

DISPLAY- “Enter elements”

for(int i=0; i<b; i=i+1)

INPUT a[i]

EndFor

int sum, d;

for (i=0; i<b; i=i+1)

sum=0,d=0;

for(int j=1; j <= a[i] / 2; j=j+1)

if (a[i]%j==0) then

sum = sum + j

EndIf

EndFor

if (sum==a[i]) then

DISPLAY- “(sum) is a perfect number”

d=d+sum AND

Bool Value of perfect=TRUE

EndIf

EndFor

if (Bool Value of perfect==TRUE) then

DISPLAY- “d is the sum of perfect numbers”

int sum1=d, digsum=0, nod=0

while (sum1! = 0) then

nod=nod+1 AND

sum1=(sum1 /10)

EndWhile

Int sum2=d

while (sum2 != 0) then

int e=sum%10 AND

digsum = digsum + pow(e, nod)

EndWhile

if (digsum==d) then

DISPLAY- “d is an Armstrong number”

EndIf

else

DISPLAY- “d is not an Armstrong number”

EndElse

EndIf

else then

DISPLAY- “There is no perfect number”

EndElse

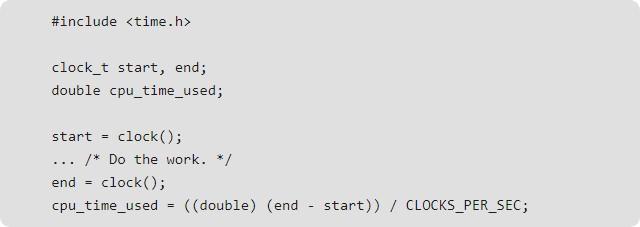
 End

# **AlGORITHM ANALYSIS**

## **Time complexity:**

The average execution time of above algorithm is calculated as **0.00481375 seconds**.

It is calculated using following syntax:



# **V. CONCLUSION**

With the help of this problem we may conclude that it becomes easy to find perfect number and armstrong number using while loop.Instead of re-writing the code again and again we only have to iterate using while loop till the condition is true.

We may also conclude that it becomes easy to store many inputs in a single variable using array. Otherwise for each input value, we may have to assign a new variable.

**VI. REFERENCES**

* <https://www.geeksforgeeks.org/program-for-armstrong-numbers/>
* <https://www.geeksforgeeks.org/perfect-number/>
* <https://www.geeksforgeeks.org/sum-factors-number/>
* ‘**Let us C’**-book by Yashwant Kanetkar

**Code for Problem Statement**

