Department of Electrical Engineering and Computer Science
Texas A&M University-Kingsville
CSEN 5303 Foundations of Computer Science
Spring 2022
Instructor: Habib M. Ammari, Ph.D. (CSE), Ph.D. (CS)
Project 1: Problem Analysis, Problem Solving, and Programming
Due Date: Feb. 17, 2022

Submitted by:

Manideep Kusa Kid# K00497313

Yeshwanth Reddy Patlolla kid#k00498214

Table of contents :

Problem Statement:

We assume that the standard input contains a sequence of non-zero integers between - 121 and 121, which ends with 0. This sequence will be given by the user.

1. Write an algorithm, called Decomposition_Powers_Three, which produces the decomposition of each integer using powers of 3, namely 1, 3, 9, 27, and 81, and the + and – operators. Each power of 3 should appear at most once in the decomposition.

Examples:

1 = 1
2 = 3 – 1
3 = 3
4 = 3 + 1
7 = 9 – 3 + 1
14 = 27 – 9 – 3 – 1
43 = 81 – 27 – 9 – 3 + 1
121 = 81 + 27 + 9 + 3 + 1

2. Show that the algorithm Decomposition_Powers_Three is correct using an informal proof (i.e., discussion).

3. Give a program corresponding to Decomposition_Powers_Three, using any of your favorite programming language

Observation: The intervals [-121,-41], [-40,-14], [-13,-5], [-4,-2], [-1,-1], [1,1], [2,4], [5,13], [14,40], and [41,121] play a particular role. To get a better understanding of our program

Discussion :

Given Number : 78

1. Subtract 1 from the given number.

2. Divide the remainder by 3, setting the quotient below the dividend and the remainder from the division to the right, whether this remainder be
2, or 1, or 0.

3. If the remainder after division is 1 or 2 proceed as directed below, but whenever the remainder after division is 0 it is necessary to subtract 1 from the quotient before proceeding to treat it as a dividend, as below.

4. Divide again by 3 as directed in step 2, and continue this process until the dividend is 0 with 0 remainder, watching throughout the process outlined in Step

5. The column of remainders 2, or 1, or 0, which have been set to the right is now numbered, beginning at the top with 0 and proceeding with 1, 2, 3, etc., and ending with the highest number in the sequence opposite the final 0 remainder. The numbers in this sequence are the powers of three.

6. Fixation of signs for the various powers, or exclusion from the statement, is determined by the remainder When the remainder is 1 the sign is negative.

When the remainder is 0 the sign is positive.
When the remainder is 1 the sign is negative.
When the remainder is 2 the power is excluded from the statement.

Demonstration of the process:

Given number : 97

	97			
	-1			
Divide by 3	96		**Power**	
	32	with remainder	0	0
	-1			
	31			
	10	with remainder	1	1
	3	with remainder	1	2
	1	with remainder	0	3
	-1			
	0		0	4

Consider the sequence from bottom to top

Remainders:	0	0	1	1	0
Signs:	+	+	-	-	+
Powers:	4	3	2	1	0

Final : $3^4 + 3^3 - 3^2 - 3^1 + 3^0$

Proof of statement:

Positive numbers: 81+27+1
Negative numbers: -9-3

Given number: 81+27+1-9-3 = 97

Algorithm:

Step1: Enter the number
Step 2: Take the input from the user and pass it as a parameter to function.
Step 3: Subtract 1 from the number and store the value in i.
Step 4: Use the while loop for continuous division of number.
Step 5: Divide the number by 3, store the remainder in x and store the quotient in i itself. If the remainder is equals to 0 subtract 1 from the i.
Step 6: Append the remainder (x) and quotient (i) to the array for each iteration of the while loop.
Step 7: Reverse the array in which remainders are stored. Let the reversed array be revrem.
Step 8: Create an array revpow with length equal to the length of remainders array and store the powers of 3 in it.
Step 9: Run the while loop till i value is greater than 0.
Step 10: Return revpow and revrem from the function.
Step 10: If remrem[i] is equal to 0 take the index position and point to the same index position in revpow and concatenate + to it after converting to string.

```
if(revrem[i]==0 ):
        z=i
        resa +='+'+str(revpow[i])
```

Step 11: If remrem[i] is equal to 1 take the index position and point to the same index position in revpow and concatenate - to it after converting to string.

```
if(revrem[i]==1):
        z=i
        resb +='-'+str(revpow[i])
```

Step 12 : Add resa , resb and store in res.
Step 13: Print res to the console.

Program using python:

```
def decompositionPower(k):

    i=k-1
    rem=[] #to store remainders of number
    quo=[] #to store quotients of number

    while(i>=0):
        if i==0:
            rem.append(0)
            quo.append(0)
            break

        x=int(i%3)  #rem
        if x==0:
            i=i-1
        i=int(i/3)  #div
        rem.append(x)
        quo.append(i)

    revrem=rem[::-1]
    # print(revrem)

    power=[]
    revpow=[]

    for j in range(0,len(revrem)):
        power.append(pow(3,j))
        revpow=power[::-1]
    # print(revpow)
    return revpow,revrem


a=list(map(int,input("enter numbers ").split()))
```

```
for i in range(0,len(a)):
    try:
        if(a[i]==0):
            print(f"{a[i]} cannot be expressed in powers of 3")
        if(a[i]>0):
            [revpow,revrem]=decompositionPower(a[i])
            resa=''
            resb=''
            z=0
            for i in range(0,len(revpow)):
                if(revrem[i]==0 ):
                    z=i
                    resa +='+'+str(revpow[i])
                if(revrem[i]==1):
                    z=i
                    resb +='-'+str(revpow[i])
                res=resa+resb
            print(res[1:])
        if(a[i]<0):
            [revpow,revrem]=decompositionPower(abs(a[i]))
            resa=''
            resb=''
            z=0
            for i in range(0,len(revpow)):
                if(revrem[i]==0 ):
                    z=i
                    resa +='-'+str(revpow[i])
                if(revrem[i]==1):
                    z=i
                    resb +='+'+str(revpow[i])
                res=resa+resb
            print(res)

    except :
        pass
```

project1.py > ...

```python

def decompositionPower(k):

    i=k-1
    rem=[] #to store remainders of number
    quo=[] #to store quotients of number

    while(i>=0):
        if i==0:
            rem.append(0)
            quo.append(0)
            break
        x=int(i%3)  #rem
        if x==0:
            i=i-1
        i=int(i/3)  #div
        rem.append(x)
        quo.append(i)
    revrem=rem[::-1]
    power=[]
    revpow=[]

    for j in range(0,len(revrem)):
        power.append(pow(3,j))
        revpow=power[::-1]

    return revpow,revrem
```

project1.py

```python
a=list(map(int,input("enter numbers ").split()))
for i in range(0,len(a)):
    try:
        if(a[i]==0):
            print(f"{a[i]} cannot be expressed in powers of 3")
        if(a[i]>0):
            [revpow,revrem]=decompositionPower(a[i])
            x=a[i]
            resa=''
            resb=''
            z=0
            for i in range(0,len(revpow)):
                if(revrem[i]==0 ):
                    z=i
                    resa +='+'+str(revpow[i])
                if(revrem[i]==1):
                    z=i
                    resb +='-'+str(revpow[i])
                res=resa+resb
            print(x ,"in decomposition powers of 3 is:",res[1:])
        if(a[i]<0):
            [revpow,revrem]=decompositionPower(abs(a[i]))
            y=a[i]
            resa=''
            resb=''
            z=0
            for i in range(0,len(revpow)):
                if(revrem[i]==0 ):
                    z=i
                    resa +=' - '+str(revpow[i])
                if(revrem[i]==1):
                    z=i
                    resb +=' + '+str(revpow[i])
                res=resa+resb
            print(y ,"in decomposition powers of 3 is:",res)
    except :
        pass
```

Output:

```
PS C:\Users\shaik\OneDrive\Desktop\Python Assignment> & C:/Users/shaik/AppData/Local/Programs/Python/Python310/python.exe "c:/Users/shaik/OneDrive/Desktop/Python Assignment/project1.py"
enter numbers 97
97 in decomposition powers of 3 is: + 81 + 27 + 1 - 9 - 3
PS C:\Users\shaik\OneDrive\Desktop\Python Assignment> & C:/Users/shaik/AppData/Local/Programs/Python/Python310/python.exe "c:/Users/shaik/OneDrive/Desktop/Python Assignment/project1.py"
enter numbers -97
-97 in decomposition powers of 3 is:  - 81 - 27 - 1 + 9 + 3
PS C:\Users\shaik\OneDrive\Desktop\Python Assignment> & C:/Users/shaik/AppData/Local/Programs/Python/Python310/python.exe "c:/Users/shaik/OneDrive/Desktop/Python Assignment/project1.py"
enter numbers 97 -97
97 in decomposition powers of 3 is: + 81 + 27 + 1 - 9 - 3
-97 in decomposition powers of 3 is:  - 81 - 27 - 1 + 9 + 3
PS C:\Users\shaik\OneDrive\Desktop\Python Assignment>
```

Conclusion:

By using this project any number can be written in powers in 3