Идея решения

Решение задачи осуществляется методом перебора вариантов расстановки знаков между цифрами. Для организации такого перебора условимся, что 0 – это отсутствие знака между цифрами, 1 – знак «+», 2 – знак «-».

В выражении может быть максимум 8 знаков, поэтому схему их расстановки логично представить как троичное число длиной в 8 цифр, каждая из которых обозначает знак (или его отсутствие) между цифрами исходного выражения. Самая левая цифра этого числа обозначает знак между первыми двумя цифрами (1 и 2) и так далее слева направо.

Для представления троичного числа выгодно воспользоваться такой структурой данных как одномерный массив, содержащий 8 элементов. Каждый из них соответствует цифре троичного числа. Чтобы перебрать все троичные числа заданной длины, достаточно организовать циклический процесс прибавления к данному числу единицы по правилам троичной арифметики до тех пор пока не наступит ситуация переполнения.

На каждом шаге этого циклического процесса требуется проверить, подходит ли данная схема расстановки знаков для выполнения условия задания или нет. Для этого используется число L = 123456789. При помощи цикла происходит перебор цифр троичного числа. Если цифра равна 0 (т.е. знака нет), то увеличивается счетчик количества цифр. Если цифра троичного числа равна 1 или 2, то выполняются следующие операции:

1. Вычисление остатка от деления L на 10 в степени, равной счетчику количества цифр. Таким образом, из L выделяется число, которое будет стоять после проверяемого знака в итоговом арифметическом выражении.
2. Выделенное число прибавляется (или вычитается, в зависимости от знака) к общей сумме (первоначально она равна 0). Кроме того, это число необходимо вычеркнуть из L. Для этого требуется вычислить результат целочисленного деления L на вышеописанную степень десятки. При дальнейшем проходе цикла L необходимо присвоить полученный результат.

Также нельзя забывать, что после последнего шага цикла останется одно неучтенное слагаемое (т.к. количество цифр больше количества знаков), поэтому в конце его необходимо прибавить к сумме. Ясно, что это будет результат последнего целочисленного деления L (см. выше).

После прохождения цикла нужно проверить, равна ли полученная сумма введенному числу. Если да, то остается только вывести текущий вариант расстановки знаков.

Вариант решения на языке C

```
#include "stdafx.h"
#include <iostream.h>
#include <conio.h>


int check(int a[8])      //Проверка троичного числа на ситуацию переполнения
{
        for (int i=0;i<=7;i++)
                if (a[i]!=2) return 0;
```

```
	return 1;
}

long work(int a[8])
{
	long sum=0,l=123456789;
	int k=0;					//Счетчик количества цифр
	for (int i=7; i>=0; i--)
	{
		int c1=1;
		k++;
		switch(a[i])		//Вычленение слагаемых из L
		{
			case 0:
				{
					break;
				}
			case 1:
				{
					for (int j=1; j<=k; j++)
						c1*=10;
					sum+=(l % c1);
					l/=c1;
					k=0;
					break;
				}
			case 2:
				{
					for (int j=1; j<=k; j++)
						c1*=10;
					sum-=(l % c1);
					l/=c1;
					k=0;
				}
		}
	}
	sum+=l;			//Прибавление последнего, неучтенного слагаемого
	return sum;
}

void output(int a[8])  // Вывод подходящей схемы
{
	cout<<1;
	for (int i=0; i<=7; i++)
	{
		switch(a[i])
		{
			case 0:
				{
					break;
				};
			case 1:
```

```
                        {
                                cout<<'+';
                                break;
                        }
                case 2:
                        {
                                cout<<'-';
                        }
        }
        cout<<i+2;
    }
}

int main()
{
    long m;
    cout << "Vvedite M ";
    cin >> m;
    int a[8];       //Троичное число, представляющее схему расстановки знаков
    for (int i=0; i<=7; i++)
    {
        a[i]=0;
    }
    a[7]=-1;
    do                  //Перебор всех вариантов
    {
        a[7]++;
        for (int i=7; i>=0; i--)
        {
            if (a[i] == 3)
            {
                a[i] = 0;
                a[i-1]++;
            }
        }
        if (work(a) == m)
        {
            output(a);
            cout<<"\n";
        }
    }
    while (check(a) != 1);
    return 0;
}
```

Вариант решения на языке Компонентный Паскаль

```
MODULE Zadacha1;
IMPORT In, StdLog;
```

```
TYPE
  ss=ARRAY 8 OF INTEGER;

PROCEDURE check(a: ss): INTEGER; (*Проверка троичного числа на ситуацию
переполнения*)
VAR
        i,flag: INTEGER;
BEGIN
        flag:=1;
        FOR i := 0 TO 7 DO
                IF a[i] # 2 THEN flag:=0;
                END;
        END;
        RETURN flag;
END check;

PROCEDURE work(a:ss): LONGINT;
VAR
        i,k,j,c1: INTEGER;
        sum,l: LONGINT;
BEGIN
        sum:=0;
        l:=123456789;
        k:=0;                                   (*Счетчик количества цифр*)
        FOR i := 7 TO 0 BY -1 DO
                c1:=1;
                k:=k+1;
                CASE a[i] OF                    (*Вычленение слагаемых из L*)
                0: ;
                |1: FOR j := 1 TO k DO
                        c1:=c1*10;
                        END;
                        sum:=sum+(l MOD c1);
                        l:= l DIV c1;
                        k:=0;
                |2:FOR j := 1 TO k DO
                        c1:=c1*10;
                        END;
                        sum:=sum-(l MOD c1);
                        l:= l DIV c1;
                        k:=0;
                END;
        END;
        sum:=sum+l;                     (*Прибавление последнего, неучтенного слагаемого*)
        RETURN sum;
END work;

PROCEDURE output(a:ss);  (*Вывод подходящей схемы расстановки знаков*)
VAR
        i: INTEGER;
BEGIN
                StdLog.String('1');
```

```
		FOR i := 0 TO 7 DO
				CASE a[i] OF
				0:;
				|1: StdLog.String('+');
				|2: StdLog.String('-');
				END;
				StdLog.Int(i+2);
			END;
END output;

PROCEDURE perebor*;
VAR
	a: ss;	(*Троичное число, представляющее схему расстановки знаков*)
	i,j: INTEGER;
	m:LONGINT;
BEGIN
	In.Open;
	In.LongInt(m);
	FOR i := 0 TO 7 DO
	 a[i] :=0;
	END;
	a[7]:=-1;
	REPEAT	(*Перебор всех вариантов*)
		INC(a[7]);
		 FOR i := 7 TO 0 BY -1 DO
			IF a[i] = 3 THEN
				a[i]:=0;
				INC(a[i-1]);
			END;
		 END;
		IF work(a) = m THEN
			output(a);
			StdLog.String('////');
		END;
	UNTIL check(a) = 1;
END perebor;

END Zadacha1.
```