```
dictionary results is a dictionarity that have every Li
        lagrnage coefficient and have the final polynomial
sm is a import of sympy from python to to represent
        variables in a polynomial
value.expand(): this function do a expands for the math
        expression (is a function from sympy)

Input: matrix data, int n
Output: dictionary results

begin lagrange

int count <- 0
array Arrx
array Arry
while (count < 2*n) do:
        if (count < n):
                Arrx.add(data[count])
        else:
                Arry.add(data[count])
end while
sizeX <- size(Arrx)
sizeY <- size(Arry)
dict result
if (sizeX != sizeY):
        x <- sm.symbols('x')
        array polynomial
        array arrayL
        count <- 0
        while (count < sizeX) do:
                int pos <- count
                float value <- Arrx[count]
                float numerator <- 1
                float denominator <- 1
                int count2 <- 1
                while (count2 < sizeX) do:
                        if count != count2:
                                numerator <- numerator*(x-Arrx[count2])
                                denominator <- denominator*(value-Arrx[count2])
                        end if
                end while
                floar aux <- numerator/denominator
                aux <- aux.expand()
                result[count] <- aux
                coefficient <- numerator*Arry[count]/denominator
                coefficient <- coefficient.expand()
                polynomial.add(coefficient)
        end while
        float sumPol <- 0
        count <- 0
        while (count < size(polynomial)):
                sumPol <- sumPol + polynomial[count]
        end while
        result["polynomial"] <- sumPol
return result

end lagrange
```