```
START
```

```
İnt PointOfWiev = 0;
String reset = query;
İnt signature[] = new int [only_Operands.length()];
```

```
getOperands while PERMUTATIONS are EXIST
```

```
String getVal = permute(query);
```

Example :
ABC,ACB,BAC,BCA,CAB,CBA

```
İnt k= 0; k<getVal.length(); k++
```

```
Char n = getVal.charAt(k);
query = Make_Zero_One(query, n);
Boolean b = calculate(query);
```

```
b == false
```

NO → `pointOfWiev ++;`

YES →
```
Signature[pointOfWiev]++;
query = reset;
pointOfWiev = 0;
```

```
FINISH
```

OUT -> LOOP

DESCRIPTION

This algorithm generates n! States for n operands and permute() method generates all permutations of operands.

Query variable equals input which is typed into by user.

Only_Operands value used to determine which priority the equation's character will be zero.

Variable pointOfWiev's initial value is zero. This variable stores numbers of how many tries till calculated query equals zero.

Reset veriable is temporary variable of query.

Make_zero_one method is makes 0 of k. Character in query string and makes all characters 1 remainder of query string.

Calculate method calculates 'AND' and 'OR' logicial operation and return false if result 0, return true if result 1.

We calculate all priority states of equation's permutations (for ABC, algorithm makes A = 0 B = 1, C = 1 and calculate equation for this values.

IF result isn't equals zero, algorithm increase pointOfWiev and tries again. (This time, algorithm makes A=0, B=0, C=1)

-> So it continues until the result is 0.

IF result == 0 , algorithm change state. (for example if ABC result == 0 algorithm pass the next state => ACB,)

This operation continues until all permutations will end, AFTER all of this calculations saved into array 'signature[]'