Article
Danny Wijnschenk · Nov 4, 2017 6m read

Advent of Code 2016 Day4: Security Through Obscurity

This is a series of programming challenges for beginners and experienced Caché programmers.

For an introduction : goto to article https://community.intersystems.com/post/advent-code-2016-day1-no-time-ta...

The input in today's challenge consists of an encrypted name, a dash, a sectorID, a dash and a checksum between brackets.
A name is real if the checksum is equal to the five most common letters in the encypted name.

For example :
aaaaa-bbb-z-y-x-123[abxyz] is a real name because the most common 5 leters are a,b,x,y,z (sorted by numer of occurences and alphabetically)

The challenge is to output the sum of the sectorID's for all the real names.

You can find the full instructions and some test data on http://adventofcode.com/2016/day/4, the input data you need to process is on http://adventofcode.com/2016/day/4/input)

Here is how the first inputlines look like :

```
gbc-frperg-pubpbyngr-znantrzrag-377[rgbnp]
nij-mywlyn-wlsiayhcw-jfumncw-alumm-mbcjjcha-422[mcjwa]
pualyuhapvuhs-ibuuf-zhslz-227[uhalp]
xlrypetn-prr-lylwjdtd-665[dzoya]
zilqwikbqdm-rmttgjmiv-mvoqvmmzqvo-278[mqvio]
rgllk-bxmefuo-sdmee-geqd-fqefuzs-274[efdgl]
```

Since every day's challenges are asking to read an input file, I decided to put that code in a separate class, and inherit from that class.

And thank to Dmitry, I changed the Open/Use/Close instructions by the more modern %Stream approach :

```
Class AOC2016.Utils
{

Parameter DIR = "J:\Winfo\AOC\";

ClassMethod Input(fileName As %String, ByRef input As %String) As %Boolean
{
  #Dim objFileStream As %Stream.FileCharacter
  #Dim status as %Boolean = 1

  Set input = 0
  Set objFileStream = ##class(%Stream.FileCharacter).%New()
```

```
  Set status = objFileStream.LinkToFile(..#DIR_fileName)
  If status'=1 Quit status
  While 'objFileStream.AtEnd {
    Set input($i(input)) = objFileStream.ReadLine()
  }
  Quit status
}

}
```

The code to calculate the checksums is not so difficult :

for each line in the input :

- count the letters in the name
- sort the letters first by occurence, than alphabetically
- take the first five letters and compare with the checksum
- if the checksum is the same, add the sectorID to the grand total

Here is the code :

```
Class AOC2016.Day4 Extends AOC2016.Utils
{

ClassMethod Part1(file As %String = "day4.txt")
{
  #Dim sumSectorID as %Integer = 0
  #Dim name, nameChecksum, checksum, letter, dummy, compareCheck, order, line,
letters, sort as %String
  #Dim iInput, sectorID, iLetter as %Integer
  #Dim input as %String

  Try {
    Do ..Input(file, .input)
    For iInput = 1:1:input {
      Set line = input(iInput)
      Set name = $Piece(line,"[",1)
      Set sectorID = $Piece(name,"-",*)
      Set nameChecksum = $Translate($Piece(name,"-",1,*-1),"-","")
      Set checksum = $Piece($Piece(line,"[",2),"]",1)
      Kill letters,sort
      // count the letters in the name :
      For iLetter=1:1:$L(nameChecksum) {
        Set letter=$E(nameChecksum,iLetter)
        Set dummy=$increment(letters(letter))
      }
      //sort the letters to get the most common
      Set letter="" For {
        Set letter=$Order(letters(letter)) If letter="" Quit
        Set sort(letters(letter),letter)=""
      }
      //take the first 5 letters
      Set compareCheck=""
      Set order="" For {
        Set order=$Order(sort(order),-1) If order="" Quit
```

```
        Set letter="" For {
          Set letter=$Order(sort(order,letter)) If letter="" Quit
          Set compareCheck = compareCheck _ letter
          If $Length(compareCheck)=5 Quit  ;we have the 5 most common letters
        }
        If $Length(compareCheck)=5 Quit
      }
      //compare with checksum and add to grand total if equal
      If compareCheck = checksum {
        Set sumSectorID = sumSectorID + sectorID
      }
    }
  } Catch {
    Write "Error : ",$ZError,!
  }
  Write "Sum of SectorIDs = ",sumSectorID,!

  Quit
}

}
```

After submitting the right answer on the adventofcode website, it unlocks the second part of the challenge :

for each real name, you need to apply a shift-cipher to decrypt the name. A shift-cipher encryption rotates all letters : e.g. A becomes B, B becomes C, .., Z becomes A and so on.
The shift-cipher will rotate each letter a fixed number of positions down the alphabet, you can find more info on this encryption technique here : https://en.wikipedia.org/wiki/Caesarcipher .

Here is the same code with the shift-cipher addition :

```
ClassMethod Part2(file As %String = "day4.txt")
{
  #Dim sumSectorID as %Integer = 0
  #Dim name, nameCipher, nameChecksum, checksum, letter, dummy,
compareCheck, order, line, letters, sort as %String
  #Dim iInput, sectorID, iLetter as %Integer
  #Dim input as %String

  Try {
    Do ..Input(file, .input)
    For iInput = 1:1:input {
      Set line = input(iInput)
      Set name = $Piece(line,"[",1)
      Set sectorID = $Piece(name,"-",*)
      Set nameCipher = $Piece(name,"-",1,*-1)
      Set nameChecksum = $Translate($Piece(name,"-",1,*-1),"-","")
      Set checksum = $Piece($Piece(line,"[",2),"]",1)
      Kill letters,sort
      For iLetter=1:1:$L(nameChecksum) {
        Set letter=$E(nameChecksum,iLetter)
        Set dummy=$i(letters(letter))
      }
      Set letter="" For {  ;sort the letters to get the most common
        Set letter=$Order(letters(letter)) If letter="" Quit
        Set sort(letters(letter),letter)=""
      }
      Set compareCheck=""
```

```
      Set order="" For {
        Set order=$Order(sort(order),-1) If order="" Quit
        Set letter="" For {
          Set letter=$Order(sort(order,letter)) If letter="" Quit
          Set compareCheck = compareCheck _ letter
          If $Length(compareCheck)=5 Quit  ;we have the 5 most common letters
        }
        If $Length(compareCheck)=5 Quit
      }
      If compareCheck = checksum {  ;room is real if checksum is correct
        Set sumSectorID = sumSectorID + sectorID
        Set nameCipher = ..ShiftCipher(nameCipher, sectorID)
        If $ZCVT(nameCipher,"l")["north" Write nameCipher," = ",sectorID,!
      }
    }
  } Catch {
    Write "Error : ",$ZError,!
  }
  Write "Sum of SectorIDs = ",sumSectorID,!

  Quit
}

ClassMethod ShiftCipher(s, times) As %String
{
  #Dim newS as %String
  #Dim i, iTimes as %Integer
  #Dim char as %Character

  //TODO : instead of rotating the string a number of times, we could also calculate
the new character directly
  For iTimes=1:1:times {
    Set newS = ""
      For i=1:1:$Length(s) {
        set char = $Extract(s,i)
        If (char=" ")!(char="-") {
        Set newS = newS_" "
      } elseIf char="Z" {
        Set newS = newS_"A"
      } elseIf char="z" {
        Set newS = newS_"a"
      } else {
        Set newS = newS _ $Char($Ascii(char)+1)
      }
    }
    Set s = newS
  }
  Quit newS
}
```

We could make the shift-cipher method quicker by eliminating one loop and directly calculating the letter after n rotations, who wants to submit this improvement ?

Look here for our solutions : https://github.com/DannyWijnschenk/AdventOfCode2016 and https://bitbucket.org/bertsarens/advent2016.

Here is the list of all Advent of Code 2016 articles :

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

#Caché #Code Snippet #Contest #ObjectScript