Questions:1

Python S3 Get Contents

In the Python file, write a program to access the contents of the bucket coderbytechallengesandbox. In there there might be multiple files, but your program should find the file with the prefix \_\_cb\_\_, and then output the contents of that file. You should use the boto3 module to solve this challenge.

You do not need any access keys to access the bucket because it is public. This post might help you with how to access the bucket.

Solution:

import requests

import boto3

from botocore import UNSIGNED

from botocore.client import Config

s3\_client = boto3.client('s3', config= Config(signature\_version=UNSIGNED))

bucket='coderbytechallengesandbox'

result = s3\_client.list\_objects(Bucket = bucket, Prefix='\_\_cb\_\_')

for content in result.get('Contents'):

    data = s3\_client.get\_object(Bucket=bucket, Key=content.get('Key'))

    contents = data['Body'].read()

    print(contents.decode("utf-8"))

question2:

In the Bash script, you will need to modify the configuration file and provide some clusters and context to it. Use the --kubeconfig flag and set it to a file named kube\_custom.config

With the CLI command config, first set 2 clusters with set-cluster. The first should be called "development", its --server should be set to "https://0.0.1.1" and the --certificate-authority should be set to "temp\_ca\_file" Then add another cluster similar to the one above, except it should be called "staging" and the server should be set to "https://5.6.7.8"

Then set 2 contexts, the first should be called "dev-frontend", the --cluser should be set to "development", the --namespace should be set to "frontend", and finally --user should be "developer". The second context should be similar to the one above, except it should be called "dev-staging" and it should be tied to the "staging" cluster.

Finally, print the contents of this config file.

#!/bin/bash

kubectl version

kubectl config set-context --current --namespace=frontend

kubectl config set-cluster development --server=https://0.0.1.1 --certificate-authority=temp\_ca\_file

kubectl config set-cluster staging --server=https://5.6.7.8

kubectl config set-context dev-frontend --cluster=development --user=developer --namespace=frontend

kubectl config set-context dev-staging --cluster=staging --user=developer --namespace=frontend

kubectl config --kubeconfig=$HOME/.kube/kube\_custom.config use-context dev-staging

export KUBECONFIG=$HOME/.kube/kube\_custom.config -n frontend

kubectl config view

question3:

Have the function BitmapHoles(strArr) take the array of strings stored in strArr, which will be a 2D matrix of 0 and 1's, and determine how many holes, or contiguous regions of 0's, exist in the matrix. A contiguous region is one where there is a connected group of 0's going in one or more of four directions: up, down, left, or right. For example: if strArr is ["10111", "10101", "11101", "11111"], then this looks like the following matrix:

1 0 1 1 1

1 0 1 0 1

1 1 1 0 1

1 1 1 1 1

For the input above, your program should return 2 because there are two separate contiguous regions of 0's, which create "holes" in the matrix. You can assume the input will not be empty.

Examples

Input: ["01111", "01101", "00011", "11110"]

Output: 3

Input: ["1011", "0010"]

Output: 2

def BitmapHoles(strArr):

  # code goes here

  return strArr

# keep this function call here

print(BitmapHoles(input()))