Cloud Computing Lab-2:

Members: Hassan Al Achek - Farid Bechaalany

Task-1:

- To create an EC2 instance, use `createEC2Instance` function which takes as arguments:
  - ec2: an instance from the Ec2Client class
  - name: a string variable which represents the EC2 instance name
  - amiId: a string variable which represents the Amazon Machine Image ID
    - Return Value:
      - The instanceId: unique id that represents the created Instance
- To switch the status of EC2 instance from running to stopped or vice-versa, use `switchStatus` function which takes as arguments:
  - ec2: an instance from the Ec2Client class
  - instanceId: a string variable that represents the created Instance unique id

Run the EC2Instance.java

- Step 1: Run the code then get the instance id from the I/O console
- Step 2: Comment the following line inside the main function to avoid the recreation of another EC2 instance:

```
String instanceId = createEC2Instance(ec2,name, amiId );
```

- Step 3: To test the `swicthStatus` function, use replace the instanceId with the one copied from step 1

```
switchStatus(ec2, instanceId);
```

Task-2:

CLASS: S3Bucket.java

- To create an S3 bucket, use `createBucket` function which takes as arguments:
  - s3: an instance from the S3Client class
  - name: a string variable that represents the s3 bucket name
- To upload a file to the S3 bucket, use `uploadFileS3Bucket` function which takes as arguments:

- s3: an instance from the S3Client class
- name: a string variable that represents the s3 bucket name
- PATH: a harcoded variable inside the code represents the path to the CSV file to be uploaded

- To create a message queue (sqs), use `createQueue` function which takes as arguments:
  - sqsClient: an instance from the SqsClient class
  - name: a string variable that represents the sqs name
- To send a message and insert it into the sqs, use `sendMessage` function which takes as arguments:
  - sqsClient: an instance from the SqsClient class
  - Sqs name: a string variable that represents the sqs name
  - PATH: a harcoded variable inside the code represents the path to the CSV file to be uploaded
  - Bucket name: a string variable that represents the S3 bucket name

Run the S3Bucket.java

- Press the run botton :)

CLASS: RetiveFileS3Bucket.java

- To get messages from SQS, use `getMessage` function which takes as arguments:
  - sqsClient: an instance from the SqsClient class
  - name: a string variable that represents the SQS name
  - Return Value:
    - A list of messages
- To delete messages from the SQS, use `deletMessages` function which takes as arguments:
  - sqsClient: an instance from the SqsClient class
  - name: a string variable that represents the SQS name
- To get a file from S3 Bucket, use `getFileS3Bucket` function which takes as arguments:
  - s3Client: an instance from the S3Client class
  - name of the S3 Bucket (names.get(0))
  - name of the file(names.get(1))
  - **Note:** This function used inside `getSum` and `minmaxValue` functions
- To get the sum, use `getSum` funtion which takes as paramters:
  - s3Client: an instance from the S3Client class
  - name of the S3 Bucket (names.get(0))
  - name of the file(names.get(1))
  - **Result**: Shown on the I/O console
- To get minimum and maximum values, use `minmaxValue` function which takes as paramters:

- s3Client: an instance from the S3Client class
- name of the S3 Bucket (names.get(0))
- name of the file(names.get(1))
- **Result**: Shown on the I/O console

- To delete the bucket content, use `deleteBucketContent` function which takes as paramters:
  - s3Client: an instance from the S3Client class
  - name of the S3 Bucket (names.get(0))
  - name of the file(names.get(1))
  - **Result**: Shown on the I/O console
- To delete the bucket itself, use `deleteS3Bucket` function which takes as paramters:
  - s3Client: an instance from the S3Client class
  - name of the S3 Bucket (names.get(0))

Run the RetiveFileS3Bucket.java

- run the code after runing the S3Bucket.java
- Press the run botton :)