**Question 1 – 7%**

Create a PVC pvc-q1 with the capacity 10Mi and storage class k8s-csi-plugin.

* Assign the PVC to the pod named my-nginx-pod with image nginx and mount to path /var/www/html and name the volume as pvc-volume.
* Ensure the pod claims the volume as ReadWriteOnce access mode
* Use kubectl patch or kubectl edit to update the capacity of the PVC as 70Mi

**Question 2- 4%**

Create a service account sa1 in new namespace ns-new.

Create a cluster role with the name new-cluster-role and ensure the role only can create below resources.

* + DaemonSets
  + Deployments
  + Replicaset

Ensure only the newly created service account can use the role and it is effective with the name space ns-new.

**Question 3- 4%**

From the pod label name=cpu-utilizer, find pods running high CPU workloads and Write the name of the pod consuming most CPU to the file /opt/KURT00101/KURT.txt.

**Question 4- 4%**

Schedule a pod as follows Name :- nginx-kusc000101 image :- nginx Node Selector :- disk=ssd.

**Question 5- 7%**

1. Create a new nginx Ingress resource as follows:
   * Name: pong
   * Namespace: ing-internal
   * Exposing service hi on path /hi using service port 5678
   * Exposing service test on path /test using service port 5678
   * The availability of service hi can be checked using the following command, which should return hi:
   * curl -kL <INTERNAL\_IP>/hi
   * The availability of service test can be checked using the following command, which should return test
   * curl -kL <INTERNAL\_IP>/test

**Question 6- 8%**

Create an NetworkPolicy named k8s-netpol in the namespace namespace-netpol1

Traffic allowed from namespace netpol2

a. Allow the pods to communicate if they are running on port 9200 from namespace netpol2  
b. Ensure the NetworkPolicy doesn’t allow other pods that are running other than port 9200

c. The communication from and to the pods running on port 9200  
d. No pods running on port 9200 from other name spaces to allowed

**Question 7- 8%**

Add a side car container to the running pod logging-pod with the blow specification

The image of the sidecar container is busybox and the container writes the logs as below tail -n+1 /var/log/k8slog/application.log

The container shares the volume logswith the application container the mounts to the directory /var/log/k8slog

Do not alter the application container and verify the logs are written properly to the file

**Question 8- 4%**

Monitor the logs of pod foo and extract log lines file-not-found. Write the output to /opt/KULM00201/foo.txt.

**Question 9- 4%**

Scale the deployment bluelable to 3 pods.

**Question 10- 4%**

Create a persistent Volume with name joe-bewada, of capactiy 2Gi and access mode ReadWriteOnce. The type of volume is hostpath and its location is /thailand/pattaya/soi7

**Question 11 – 8%**

Save the snapshot of the ETC cluster https://120.0.0.1:2379 to the location /opt/data/backup/etcd-backup-snapshot.db  
◼**Note**: The ETCD back up will be completed less than minute and if that is taking more time , do not mind to press Ctrl+C.

⚫ Restore the ETCD using the backup taken previously at /opt/data/backup/etcd-backup-snapshot-previous.db

* Use the below certificates to configure the cluster

--ca-file=/opt/KUCM0032/ca.crt

--certfile=/opt/KUCM00302/etcd-client.crt

--key-file=/opt/KUCM00302/etcd-client.key

**Question 12 – 8%**

Upgrade master control plane components from version 1.18.1 to only version 1.18.2

* Drain the master before start the upgrade and uncordn once the upgrade is completed
* Update the kubelet and kubeadm as well  
  ***Note : Do not touch and make any changes on the Worker node***

**Question 13 – 4%**

The node k8s-node-01 is not in ready state. Ssh to the node and troubleshoot the issue

systemctl start kubelet  
systemctl list-unit-files | grep kubelet

sudo systemctl list-unit-files

**Question 14 – 4%**

Check to see how many nodes are ready (not including nodes tainted NoSchedule) and write the number to /opt/KUCC00104/kucc00104.txt.

kubectl describe nodes | grep -i taint | grep -v NoSchedule | wc -l > kucc00104.txt

kubectl get nodes -o=jsonpath='{.items[\*].spec.taints[?(@.effect == "NoSchedule")]}{"\n"}' | wc -l

kubectl describe nodes | grep -i taint | grep -v NoSchedule | wc -l

**Question 15 – 7%**

The pod mango is running in name space mango-ns and is in pending state. Fix the issue. ***You may ssh to the node that the pod is pending to schedule and troubleshoot.***

**Question 16 – 4%**

Create a pod named kucc8 with a single app container for each of the following images running inside (there may be between 1 and 4 images specified): nginx + redis + memcached + consul.

**Question 17 – 4%**

Set the node names ek8s-node-0 as unavailable & reschedule all the pods running on it.

kubectl get pods -A -o wide | grep shobhans2c.mylabserver.com

kubectl drain shobhans2c.mylabserver.com --ignore-daemonsets

**Question 18 – 7%**

* Reconfigure the existing deployment front-end:  
  Add a port specification named exposing http port 80/tcp of the existing container nginx add a selector entry with key foo and value bar
* Create a new service named front-end-svc exposing the container port http. Also use the new selector entry foo to select the service's target.
* Configure the new service to also expose the individual Pods via a port on the host on which they are scheduled.