1. Set configuration context $ kubectl config use-context k8s

Monitor the logs of Pod foobar and

- Extract log lines corresponding to error unable-to-access-website
- Write them to /opt/KULM00201/foobar

Question weight 5%

Answer:

$ kubectl logs pod foobar | grep “ unable-to-access-website ” > /opt/KULM00201/foobar

2. Set configuration context $ kubectl config use-context k8s

List all PVs sorted by **name**, saving the full kubectl output to /opt/KUCC0010/my_volumes . Use kubectl s own functionally for sorting the output, and do not manipulate it any further.

Question weight 3%

Answer：

$kubectl get pv --all-namespaces --sort-by="metadata.name" >> /opt/KUCC0010/my_volumes **or**

$kubectl get pv --all-namespaces --sort-by="spec.capacity.storage" >> /opt/KUCC0010/my_volumes

3. Set configuration context $ kubectl config use-context k8s

Ensure a single instance of Pod nginx is running on each node of the kubernetes cluster where nginx also represents the image name which has to be used. Do no override any taints currently in place.

Use **Daemon sets** to complete this task and use ds.kusc00201 as Daemonset name.

Question weight 3%

Answer：

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: ds.kusc...
spec:
  selector:
    matchLabels:
      name: nginx
  template:
    metadata:
      labels:
        name: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
```

4. Set configuration context `$ kubectl config use-context k8s`

Perform the following tasks

- Add an init container to lumpy--koala (Which has been defined in spec file /opt/kucc00100/pod-spec-KUCC00100.yaml)
- The init container should create an empty file named /workdir/calm.txt
- If /workdir/calm.txt is not detected, the Pod should exit
- Once the spec file has been updated with the init container definition, the Pod should be created.

Question weight 7%

yaml 文件：

```
apiVersion： v1
kind： Pod
Metadate:
    Name lumpy-koala
Spec
    Volumes
        - name workdir
          Emptydir
    Containers
        - name checker
          Image alpine
          Command    .......
        Volumemount
          - name workdir
            Mountpath /workdir
```

Answer:

```
apiVersion: v1
kind: pod
metadata:
  name: lumpy-koala
```

```
spec:
  volumes:
    - name: workdir
      emptyDir: {}
  contianers:
    - name: checker
      image: alpine
      command: ....
      volumeMount:
        - name: workdir
          mountPath: /workdir
```

```
initContainers:
    - name: init-lumpy-koala
      image: busybox
      command: ['sh','-c','touch /workdir/calm.txt']
      volumeMounts:
        - name: workdir
          mountPath: "/workdir"
```

```
kubectl   create   -f   opt/kucc00100/pod-spec-KUCC00100.yaml
```

5. Set configuration context $ kubectl config use-context k8s
Create a pod named kucc4 with a single container for each of the following images running inside (there may be between 1 and 4 images specified): nginx + redis + memcached + consul
Question weight: 4%

Answer:

```
apiVersion: v1
kind: Pod
metadata:
    name: kucc4
spec:
    containers:
        - name: nginx
          image: nginx
    containers:
        - name: redis
          image: redis
    containers:
        - name: memcached
          image: memcached
    containers:
        - name: consul
          image: consul
```

```
kubectl create -f kucc4.yaml
```

6. Set configuration context $ kubectl config use-context k8s

Schedule a Pod as follows:

- Name: nginx-kusc00101
- Image: nginx
- Node selector: disk=ssd

Question weight: 2%

Answer:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-kusc00101
spec:
  containers:
    - name: nginx-kusc00101
      image: nginx
  nodeSelector:
    disk: ssd
```

```
kubectl create -f nginx-kusc00101.yaml
```

7. Set configuration context $ kubectl config use-context k8s

Create a deployment as follows

- Name: nginx-app
- Using container nginx with version 1.11.9-alpine
- The deployment should contain 3 replicas

Next, deploy the app with new version 1.12.0-alpine by performing a rolling update and record that update.

Finally, rollback that update to the previous version 1.11.9-alpine

Question weight: 4%

Answer:

```
kubectl run nginx-app --image='nginx:1.11.9-alpine'
--replicas=3
```

```
kubectl set image deployment/nginx-app
nginx-app='nginx:1.12.0-alpine' --record=true
```

```
kubectl rollout undo deployment/nginx-app
```

8. Set configuration context $ kubectl config use-context k8s
Create and configure the service front-end-service so it's accessible through NodePort / ClusterIP and routes to the existing pod named front-end
Question weight: 4%

Answer:

```
kubectl expose pod front-end --name=front-end-service
--type="NodePort" --port=80
```

Or

9. Set configuration context $ kubectl config use-context k8s
Create a Pod as follows:

- Name: jenkins
- Using image: jenkins
- In a new Kubenetes namespace named website-frontend

Question weight 3%

Answer:

```
kubectl create ns website-frontend
```

```
apiVersion: v1
kind: Pod
metadata:
        name: jenkins
        namespace: website-frontend
spec:
        containers:
                - name: jenkins
                  image: jenkins
```

```
kubectl create -f jenkins.yaml
```

10. Set configuration context $ kubectl config use-context k8s

Create a deployment spec file that will:

- Launch 7 replicas of the redis image with the label: app_env_stage=dev
- Deployment name: kual00201

Save a copy of this spec file to /opt/KUAL00201/deploy_spec.yaml (or .json)

When you are done, clean up (delete) any new k8s API objects that you produced during this task

Question weight: 3%

Answer:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kual00201
spec:
  replicas: 7
  selector:
    matchLabels:
      app_env_stage: dev
  template:
    metadata:
      labels:
        app_env_stage: dev
    spec:
      containers:
      - name: redis
        image: redis
kubectl create -f kual00201.yaml
cp kual00201.yaml /opt/KUAL00201/deploy_spec.yaml
kubectl delete deploy kual00201
```

11. Set configuration context $ kubectl config use-context k8s

Create a file /opt/KUCC00302/kucc00302.txt that lists all pods that implement Service **foo** in Namespace **production**.

The format of the file should be one pod name per line.

Question weight: 3%

Answer:

```
kubectl get svc -o wide   -n production
```

得到 label

```
kubectl get pod -l label=value |awk '{print $1}' |grep -v "NAME" >> /opt/KUCC00302/kucc...
kubectl get po -l label
```

12. Set configuration context $ kubectl config use-context k8s

Create a Kubernetes Secret as follows:

- Name: super-secret
- Credential: alice **or** username: bob

Create a Pod named pod-secrets-via-file using the **redis** image which mounts a secret named super-secret at /secrets

Create a second Pod named pod-secrets-via-env using the **redis** image, which exports **credential** / **username** as TOPSECRET / CREDENTIALS

Question weight: 9%

Answer:

```
kubectl create secret generic spuer-secret
--from-literal=Credential=alice
```

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-secrets-via-file
spec:
  containers:
  - name: pod-secrets-via-file
    image: redis
    volumeMounts:
    - name: super-secret
      mountPath: "/secret"
  volumes:
  - name: super-secret
    secret:
      secretName: super-secret
```

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-secrets-via-env
spec:
  containers:
  - name: pod-secrets-via-env
    image: redis
    env:
```

```
    - name: TOPSECRET
      valueFrom:
        secretKeyRef:
          name: super-secret
          key: Credential
```

13. Set configuration context $ kubectl config use-context k8s
Create a pad as follows:

- Name: non-persistent-redis
- Container image: redis
- Named-volum with name: cache-control
- Mount path: /data/redis

It should launch in the pre-prod namespace and the volume MUST NOT be persistent.
Question weight: 4%

Answer:

```
apiVersion: v1
kind: Pod
metadata:
  name: non-persistent-redis
  namespace: pre-prod
spec:
  containers:
  - name: non-persistent-redis
    image: redis
    volumeMounts:
    - name: cache-control
      mountPath: "/data/redis"
  volumes:
  - name: cache-control
    emptyDir: {}
```

14. Set configuration context $ kubectl config use-context k8s
Scale the deployment **webserver** to **6** pods
Question weight: 1%

Answer:

```
kubectl scale deployment/webserver --replicas=6
```

15. Set configuration context $ kubectl config use-context k8s
Check to see how many nodes are ready (not including nodes tainted NoSchedule) and write the

number to /opt/......
Question weight: 2%

Answer: 2

首先用 **kubectl get node** 查看总共 node 数量，然后 **kubectl describe node** 查看不包含污点值为 NoSchedule 的 node 数量

16. Set configuration context $ kubectl config use-context k8s
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/...... (which already exists)
Question weight: 2%

Answer:

通过 **kubectl get pod -l name=cpu-utilizer -o wide** 查看 pod 所在的 node

之后 **kubectl describe node** 查看哪个 pod 请求的 cpu 最多

三个都是 0，我就都写进去了

17. Set configuration context $ kubectl config use-context k8s
Create a deployment as follows

- Name: nginx-dns
- Exposed via a service: nginx-dns
- Ensure that the service & pod are accessible via their respective DNS records
- The container(s) within any Pod(s) running as a part of this deployment should use the **nginx** image

Next, use the utility nslookup to look up the DNS records of the service & pod and write the output to /opt/..../service.dns and /opt/..../pod.dns respectively. Ensure you use the busybox:1.28 image (or earlier) for any testing, an the latest release has an unpstream bug which impacts the use of nslookup.
Question weight: 7%

Answer:

kubectl run nginx-dns --image=nginx:latest

kubectl expose deployment/nginx-dns --port=80 --name=nginx-dns

kubectl run busybox -it --rm --image=busybox sh

nslookup nginx-dns #svc 记录

nslookup pod-id-addr #pod 记录

第四行：　是查 service 的 dns
第五行：　是查 pod 的
Nslookup 的结果不能直接输出到文件，我是粘贴到 notepad 上再用

18. No configuration context change required for this item
Create a snapshot of the etcd instance running at https://127.0.0.1:2379 saving the snapshot to the file path /data/backup/etcd-snapshot.db
The etcd instance is running etcd version 3.2.18
The following TLS certificates/key are supplied for connecting to the server with etcdctl

- CA certificate: /opt/KUCM00302/ca.crt
- Client certificate: /opt/KUCM00302/etcd-client.crt
- Client key: /opt/KUCM00302/etcd-client.key

Question weight: 7%

Answer：

```
$ export ETCDCTL_API=3

$ etcdctl help

$ etcdctl --endpoints=https://127.0.0.1:2379  \

--ca-file=/opt/KUCM00302/ca.crt  \

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

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

snapshot save /data/backup/etcd-snapshot.db
```

因为 etcd 版本不一样，后面关键字跟的方法也不一样，先 help 看一下，照着帮助写就没问题。

19. Set configuration context $ kubectl config use-context ek8s
Set the node labelled with name=ek8s-node-1 as unavailable and reschedule all the pods running on it.
Question weight: 4%

Answer:

```
kubectl get node --show-labels |grep name=ek8s-node-1   #找出 node

kubectl drain node $node
```

20. Set configuration context $ kubectl config use-context wk8s

A Kubernetes worker node, labelled with **name=wk8s-node-0** is in state NotReady . Investigate why this is the case, and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made permanent.

Hints:

- You can ssh to the failed node using $ ssh wk8s-node-0
- You can assume elevated privileges on the node with the following command $ sudo -i

Question weight: 4%

Answer:

是 kubelet 没有启动

kubectl get node

查看 一个 node 是 notReady

ssh 上去

systemctl status kubelet

发现没有启动

systemctl start kubelet

21. Set configuration context $ kubectl config use-context wk8s

Configure the kubelet systemd managed service, on the node labelled with **name=wk8s-node-1**, to launch a Pod containing a single container of image nginx named myservice automatically. Any spec files required should be placed in the /etc/kubernetes/manifests directory on the node.

Hints:

- You can ssh to the failed node using $ ssh wk8s-node-1
- You can assume elevated privileges on the node with the following command $ sudo -i

Question weight: 4%

Answer

1. 登陆到节点上

找到 kubelet 的目录

/etc/kubenetes/manifests/

创建下面的 yaml ， 并保存在这个目录

```
apiVersion: v1
kind: Pod
metadata:
  name: webtool
spec:
  containers:
  - name: myservice
    image: nginx
```

2. vi /etc/systemctl/system/kubelet.service

在 /usr/bin/kubelete 下面第一行加入
--pod-manifest-path=/etc/kubenetes/manifests/ \

保存

```
systemctl daemon-reload

systemctl restart kubelet


docker ps | grep webtool
```

22. Set configuration context $ kubectl config use-context ik8s
In this task, you will configure a new Node, **ik8s-node-0**, to join a Kubernetes cluster as follows:

- Configure kubelet for automatic certificate rotation and ensure that both server and client CSRs are automatically approved and signed as appropnate via the use of RBAC.
- Ensure that the appropriate cluster-info ConfigMap is created and configured appropriately in the correct namespace so that future Nodes can easily join the cluster
- Your bootstrap kubeconfig should be created on the new Node at /etc/kubernetes/bootstrap-kubelet.conf (do not remove this file once your Node has successfully joined the cluster)
- The appropriate cluster-wide CA certificate is located on the Node at /etc/kubernetes/pki/ca.crt . You should ensure that any automatically issued certificates are installed to the node at /var/lib/kubelet/pki and that the kubeconfig file for kubelet will be rendered at /etc/kubernetes/kubelet.conf upon successful bootstrapping
- Use an additional group for bootstrapping Nodes attempting to join the cluster which should be called system:bootstrappers:cka:default-node-token
- Solution should start automatically on boot, with the systemd service unit file for kubelet available at /etc/systemd/system/kubelet.service

To test your solution, create the appropriate resources from the spec file located at /opt/..../kube-flannel.yaml This will create the necessary supporting resources as well as the kube-flannel -ds DaemonSet . You should ensure that this DaemonSet is correctly deployed to the single node in the cluster.
Hints:

- kubelet is not configured or running on **ik8s-master-0** for this task, and you should not attempt to configure it.

- You will make use of TLS bootstrapping to complete this task.
- You can obtain the IP address of the Kubernetes API server via the following command $ ssh ik8s-node-0 getent hosts ik8s-master-0
- The API server is listening on the usual port, 6443/tcp, and will only server TLS requests
- The kubelet binary is already installed on **ik8s-node-0** at /usr/bin/kubelet . You will not need to deploy kube-proxy to the cluster during this task.
- You can ssh to the new worker node using $ ssh ik8s-node-0
- You can ssh to the master node with the following command $ ssh ik8s-master-0
- No further configuration of control plane services running on **ik8s-master-0** is required
- You can assume elevated privileges on both nodes with the following command $ sudo -i
- Docker is already installed and running on **ik8s-node-0**

Question weight: 8%

23. Set configuration context $ kubectl config use-context bk8s

Given a partially-functioning Kubenetes cluster, identify symptoms of failure on the cluster. Determine the node, the failing service and take actions to bring up the failed service and restore the health of the cluster. Ensure that any changes are made permanently.

The worker node in this cluster is labelled with **name=bk8s-node-0**

Hints:

- You can ssh to the relevant nodes using $ ssh $(NODE) where $(NODE) is one of **bk8s-master-0** or **bk8s-node-0**
- You can assume elevated privileges on any node in the cluster with the following command $ sudo -i

Question weight: 4%

是 kube-manager-controller 没有启动 启动就做完了

kc get cs
能看到 controller manager 没有启动
登陆到 master 上

systemctl start kube-manager-controller.service # 到机器上的 /etc/systemd/system 确认一下是不是这么写的

24. Set configuration context $ kubectl config use-context hk8s

Creae a persistent volume with name app-config of capacity 1Gi and access mode ReadWriteOnce. The type of volume is hostPath and its location is /srv/app-config

Question weight: 3%

Answer:

apiVersion: v1
kind: PersistentVolume

```yaml
metadata:
  name: app-config
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /srv/app-config
```