My Exam Prep:

* CNCF Kubernetes Class + labs
* K8S The Hard Way run through
* Run through all the tasks in the k8s docs
* Practice with systemd, journald, openssl, cfssl, and etcd
* Work through the sections in Walid’s github list

Try the following exercises interactively:

Note - there are no answers here on purpose. You should be able to do these yourself using the minimal docs that you are allowed to use during the test. At a minimum this should train you on where to look for this info during the test, without notes.

1. Create a node that has a SSD and label it as such.
2. Create a pod that is only scheduled on SSD nodes.

<https://github.com/kubernetes/website/blob/master/content/en/docs/tasks/configure-pod-container/assign-pods-nodes.md>

1. Create 2 pod definitions: the second pod should be scheduled to run anywhere the first pod is running - 2nd pod runs alongside the first pod.

<https://github.com/dgkanatsios/CKAD-exercises/blob/master/b.multi_container_pods.md>

1. Create a deployment running nginx version 1.12.2 that will run in 2 pods  
    a. Scale this to 4 pods.  
    b. Scale it back to 2 pods.  
    c. Upgrade this to 1.13.8  
    d. Check the status of the upgrade  
    e. How do you do this in a way that you can see history of what happened?  
    f. Undo the upgrade

<https://github.com/dgkanatsios/CKAD-exercises/blob/master/c.pod_design.md>

1. Create a service that uses a scratch disk.  
    a. Change the service to mount a disk from the host.  
    b. Change the service to mount a persistent volume.

<https://github.com/wonkarthik/k8s-questions/blob/master/k8s-exercise/g.state.md>

1. Create a pod that has a liveness check

<https://github.com/wonkarthik/k8s-questions/blob/master/practice-questions-with-solutions.md>

1. Create a service that manually requires endpoint creation - and create that too

<https://github.com/wonkarthik/k8s-questions/blob/master/k8s-exercise/f.services.md>

1. Create a daemon set  
    a. Change the update strategy to do a rolling update but delaying 30 seconds between pod updates
2. Create a static pod

<https://suraj.pro/post/static-pod/>

<https://github.com/kubernetes/website/blob/master/content/en/docs/tasks/administer-cluster/static-pod.md>

<https://medium.com/@sonasingh46/static-pod-in-kubernetes-e3854507655f>

https://medium.com/mayadata/story-of-pod-manifest-to-running-8e4b38f074ec

1. Create a busybox container without a manifest. Then edit the manifest.
2. Create a pod that uses secrets  
    a. Pull secrets from environment variables  
    b. Pull secrets from a volume  
    c. Dump the secrets out via kubectl to show it worked

<https://github.com/wonkarthik/k8s-questions/blob/master/k8s-exercise/d.configuration.md>

1. Create a job that runs every 3 minutes and prints out the current time.

<https://github.com/wonkarthik/k8s-questions/blob/master/k8s-exercise/c.pod_design.md>

--schedule="\*/3 \* \* \* \*"

kubectl run busybox --image=busybox --restart=OnFailure --schedule="\*/3 \* \* \* \*" -- /bin/sh -c 'date’

1. Create a job that runs 20 times, 5 containers at a time, and prints "Hello parallel world"

<https://github.com/wonkarthik/k8s-questions/blob/master/k8s-exercise/c.pod_design.md>

completions: 20

parallelism: 5

kubectl run busybox --image=busybox --restart=OnFailure --dry-run -o yaml -- /bin/sh -c 'echo Hello parallel world' > job.yaml

1. Create a service that uses an external load balancer and points to a 3 pod cluster running nginx.
2. Create a horizontal autoscaling group that starts with 2 pods and scales when CPU usage is over 50%.

kubectl autoscale deploy nginx --min=2 --max=10 --cpu-percent=50

1. Create a custom resource definition  
    a. Display it in the API with curl
2. Create a networking policy such that only pods with the label access=granted can talk to it.  
    a. Create an nginx pod and attach this policy to it.   
    b. Create a busybox pod and attempt to talk to nginx - should be blocked  
    c. Attach the label to busybox and try again - should be allowed

<https://github.com/dgkanatsios/CKAD-exercises/blob/master/f.services.md>

1. Create a service that references an externalname.  
    a. Test that this works from another pod
2. Create a pod that runs all processes as user 1000.

<https://github.com/dgkanatsios/CKAD-exercises/blob/master/d.configuration.md>

1. Create a namespace  
    a. Run a pod in the new namespace  
    b. Put memory limits on the namespace  
    c. Limit pods to 2 persistent volumes in this namespace

<https://github.com/KrutikaShah52/Kubernetes_Course/blob/master/namespace/resourcequota.yaml>

1. Write an ingress rule that redirects calls to /foo to one service and to /bar to another

<https://github.com/savicprvoslav/kubernetes-front-end-backend-example>

https://github.com/devopstraining4/kubernetes\_ingress\_example

1. Write a service that exposes nginx on a nodeport  
    a. Change it to use a cluster port  
    b. Scale the service  
    c. Change it to use an external IP  
    d. Change it to use a load balancer
2. Deploy nginx with 3 replicas and then expose a port  
    a. Use port forwarding to talk to a specific port
3. Make an API call using CURL and proper certs

curl --cacert /etc/kubernetes/pki/ca.crt --key /etc/kubernetes/ pki/apiserver-kubelet-client.key --cer

1. Upgrade a cluster with kubeadm

<https://blog.51cto.com/ygqygq2/2366609>

1. Get logs for a pod

Kubectl logs podname

1. Deploy a pod with the wrong image name (like --image=nginy) and find the error message.

kubectl create deployment my-nginx --image=nginy

kubectl get events

1. Get logs for kubectl
2. Get logs for the scheduler
3. Restart kubelet  
     
   **Non-K8S**
4. Convert a CRT to a PEM  
    a. Convert it back
5. Backup an etcd cluster
6. List the members of an etcd cluster
7. Find the health of etcd

https://github.com/HanshuLin/KubernetesExcercise/blob/master/read