



Linux-Foundation

Exam Questions CKAD

Certified Kubernetes Application Developer (CKAD) Program

About Exambible

[Your Partner of IT Exam](#)

Found in 1998

Exambible is a company specialized on providing high quality IT exam practice study materials, especially Cisco CCNA, CCDA, CCNP, CCIE, Checkpoint CCSE, CompTIA A+, Network+ certification practice exams and so on. We guarantee that the candidates will not only pass any IT exam at the first attempt but also get profound understanding about the certificates they have got. There are so many alike companies in this industry, however, Exambible has its unique advantages that other companies could not achieve.

Our Advances

* 99.9% Uptime

All examinations will be up to date.

* 24/7 Quality Support

We will provide service round the clock.

* 100% Pass Rate

Our guarantee that you will pass the exam.

* Unique Gurantee

If you do not pass the exam at the first time, we will not only arrange FULL REFUND for you, but also provide you another exam of your claim, ABSOLUTELY FREE!

NEW QUESTION 1

Exhibit:



Context

It is always useful to look at the resources your applications are consuming in a cluster. Task

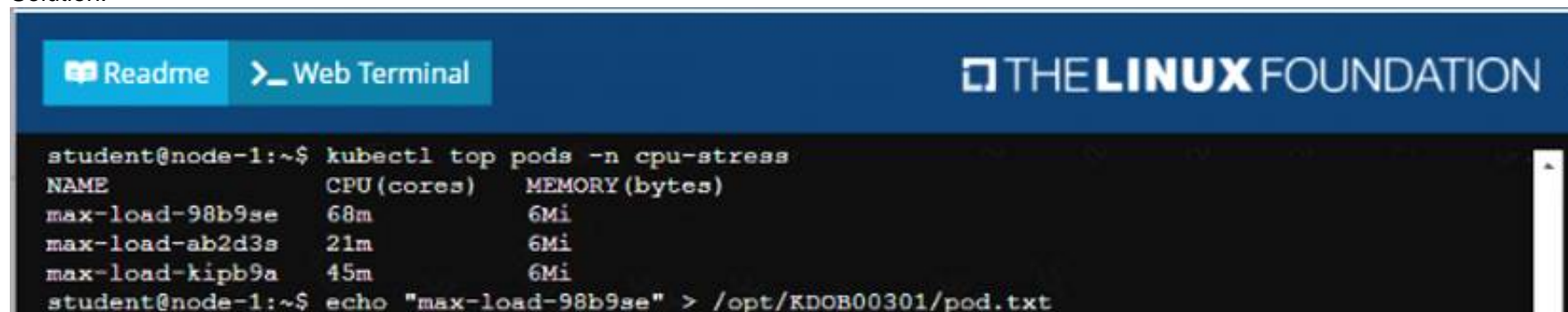
- From the pods running in namespacecpu-stress , write the name only of the pod that is consuming the most CPU to file /opt/KDOBG030I/pod.txt, which has already been created.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



NEW QUESTION 2

Exhibit:



Context

You sometimes need to observe a pod's logs, and write those logs to a file for further analysis. Task

Please complete the following;

- Deploy the counter pod to the cluster using the provided YAMLSpec file at /opt/KDOB00201/counter.yaml
- Retrieve all currently available application logs from the running pod and store them in the file /opt/KDOB0020I/log_Output.txt, which has already been created

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
student@node-1:~$ kubectl create -f /opt/KDOB00201/counter.yaml
pod/counter created
student@node-1:~$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
counter             1/1     Running   0           10s
liveness-http       1/1     Running   0           6h45m
nginx-101           1/1     Running   0           6h46m
nginx-configmap     1/1     Running   0           107s
nginx-secret        1/1     Running   0           7m21s
poller              1/1     Running   0           6h46m
student@node-1:~$ kubectl logs counter
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbe5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$
```

```
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
```

Readme

Web Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbe5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbc1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3fbe74
17: 9ae01bef01748b12cc9f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$
```

NEW QUESTION 3

Exhibit:



Context

A user has reported an application is unteachable due to a failing livenessProbe . Task

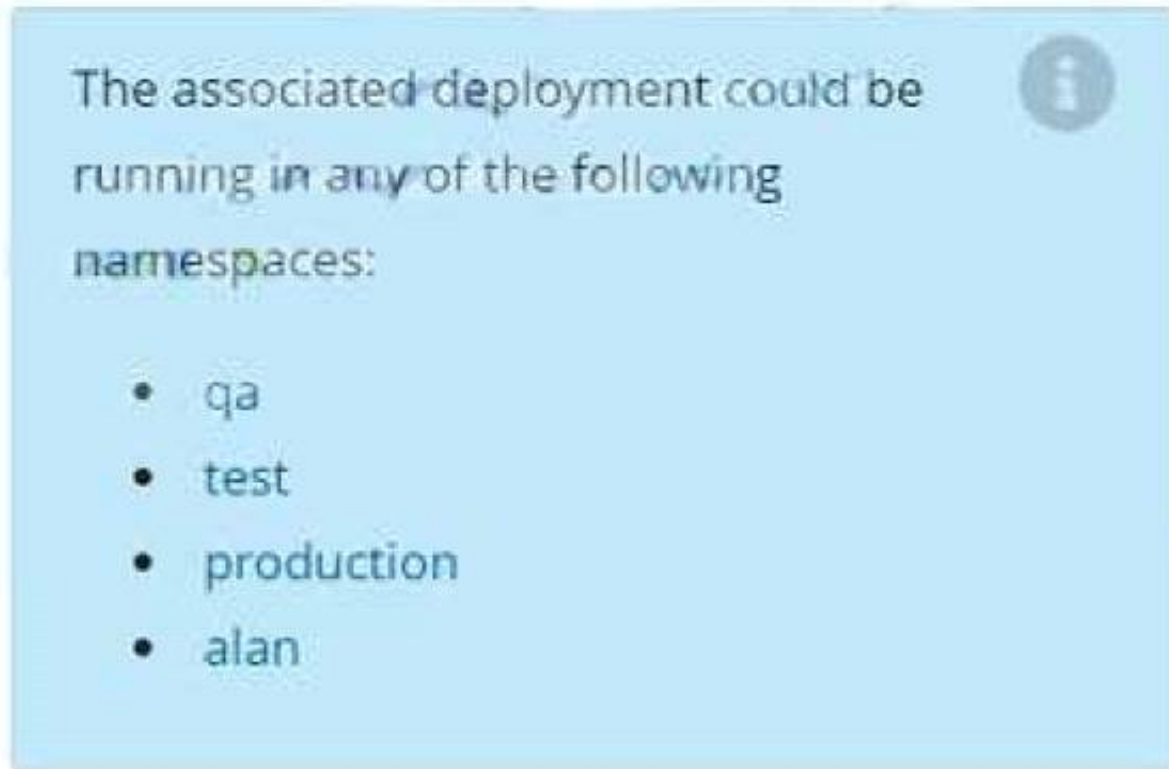
Perform the following tasks:

- Find the broken pod and store its name and namespace to /opt/KDOB00401/broken.txt in the format:

```
<namespace>/<pod>
```


The output file has already been created

- Store the associated error events to a file /opt/KDOB00401/error.txt, The output file has already been created. You will need to use the -o wide output specifier with your command
- Fix the issue.



- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

Create the Pod: `kubectlcreate-f`

`http://k8s.io/docs/tasks/configure-pod-container/`

`exec-liveness.yaml`

Within 30 seconds, view the Pod events: `kubectldescribepod liveness-exec`

The output indicates that no liveness probes have failed yet:

FirstSeen LastSeen CountFrom SubobjectPath Type Reason Message

24s 24s 1{default-scheduler } NormalScheduled Successfully assigned liveness-exec to worker0

23s 23s 1{kubelet worker0} spec.containers{liveness} NormalPulling pulling image"gcr.io/google_containers/busybox"

23s 23s 1{kubelet worker0} spec.containers{liveness} NormalPulled Successfully pulled image"gcr.io/google_containers/busybox"

23s 23s 1{kubelet worker0} spec.containers{liveness} NormalCreated Created container with docker id86849c15382e; Security:[seccomp=unconfined]

23s 23s 1{kubelet worker0} spec.containers{liveness} NormalStarted Started container with docker id86849c15382e

After 35 seconds, view the Pod events again: `kubectldescribepod liveness-exec`

At the bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been killed and recreated.

FirstSeen LastSeen Count From SubobjectPath Type Reason Message

37s 37s 1{default-scheduler } Normal Scheduled Successfully assigned liveness-exectoworker0

36s 36s 1{kubelet worker0} spec.containers{liveness} Normal Pulling pulling image"gcr.io/google_containers/busybox"

36s 36s 1{kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image"gcr.io/google_containers/busybox"

36s 36s 1{kubelet worker0} spec.containers{liveness} Normal Created Created containerwithdocker id86849c15382e; Security:[seccomp=unconfined]

36s 36s 1{kubelet worker0} spec.containers{liveness} Normal Started Started containerwithdocker id86849c15382e

2s 2s 1{kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open

'/tmp/healthy': No suchfileordirectory

Wait another 30 seconds, and verify that the Container has been restarted: `kubectl get pod liveness-exec`

The output shows that RESTARTS has been incremented:

NAMEREADY STATUSRESTARTS AGE

liveness-exec 1/1Running 1m

NEW QUESTION 4

Exhibit:



Task

You are required to create a pod that requests a certain amount of CPU and memory, so it gets scheduled to-a node that has those resources available.

- Create a pod named `nginx-resources` in the `pod-resources` namespace that requests a minimum of 200m CPU and 1Gi memory for its container

- The pod should use the nginx image
- The pod-resources namespace has already been created

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Solution:

[illegible]

Readme

Web Terminal

THE **LINUX** FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-resources
    name: nginx-resources
    namespace: pod-resources
spec:
  containers:
  - image: nginx
    name: nginx-resources
    resources:
      requests:
        cpu: 200m
        memory: "1Gi"
```

-- INSERT --

15,22All

ReadmeWeb Terminal

THE **LINUX** FOUNDATION

```
student@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx --dry-run=client -o
yaml > nginx_resources.yml
student@node-1:~$ vim nginx_resources.yml
student@node-1:~$ kubectl create -g nginx_resources.yml
Error: unknown shorthand flag: 'g' in -g
See 'kubectl create --help' for usage.
student@node-1:~$ kubectl create -f nginx_resources.yml
pod/nginx-resources created
student@node-1:~$ kubectl get pods -n pod-re
```

ReadmeWeb Terminal

THE **LINUX** FOUNDATION

```
student@node-1:~$ kubectl get pods -n pod-resources
NAME          READY   STATUS    RESTARTS   AGE
nginx-resources 1/1     Running   0           8s
student@node-1:~$
```

NEW QUESTION 5

Exhibit:



Context

A web application requires a specific version of redis to be used as a cache. Task

Create a pod with the following characteristics, and leave it running when complete:

- The pod must run in the web namespace. The namespace has already been created
- The name of the pod should be cache
- Use the lfcncf/redis image with the 3.2 tag
- Expose port 6379

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

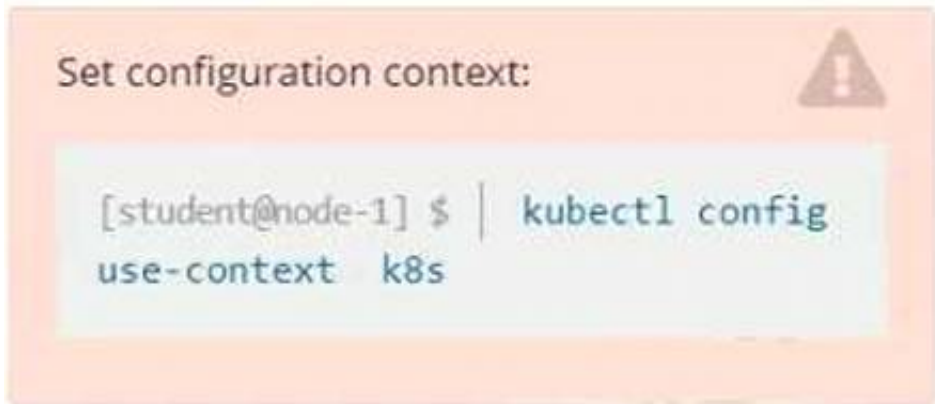
ReadmeWeb Terminal

THE **LINUX** FOUNDATION

```
student@node-1:~$ kubectl run cache --image=lfcncf/redis:3.2 --port=6379 -n web
pod/cache created
student@node-1:~$ kubectl get pods -n web
NAME    READY   STATUS             RESTARTS   AGE
cache   0/1     ContainerCreating   0           6s
student@node-1:~$ kubectl get pods -n web
NAME    READY   STATUS    RESTARTS   AGE
cache   1/1     Running   0           9s
student@node-1:~$
```

NEW QUESTION 6

Exhibit:



Context

Your application's namespace requires a specific service account to be used.

Task

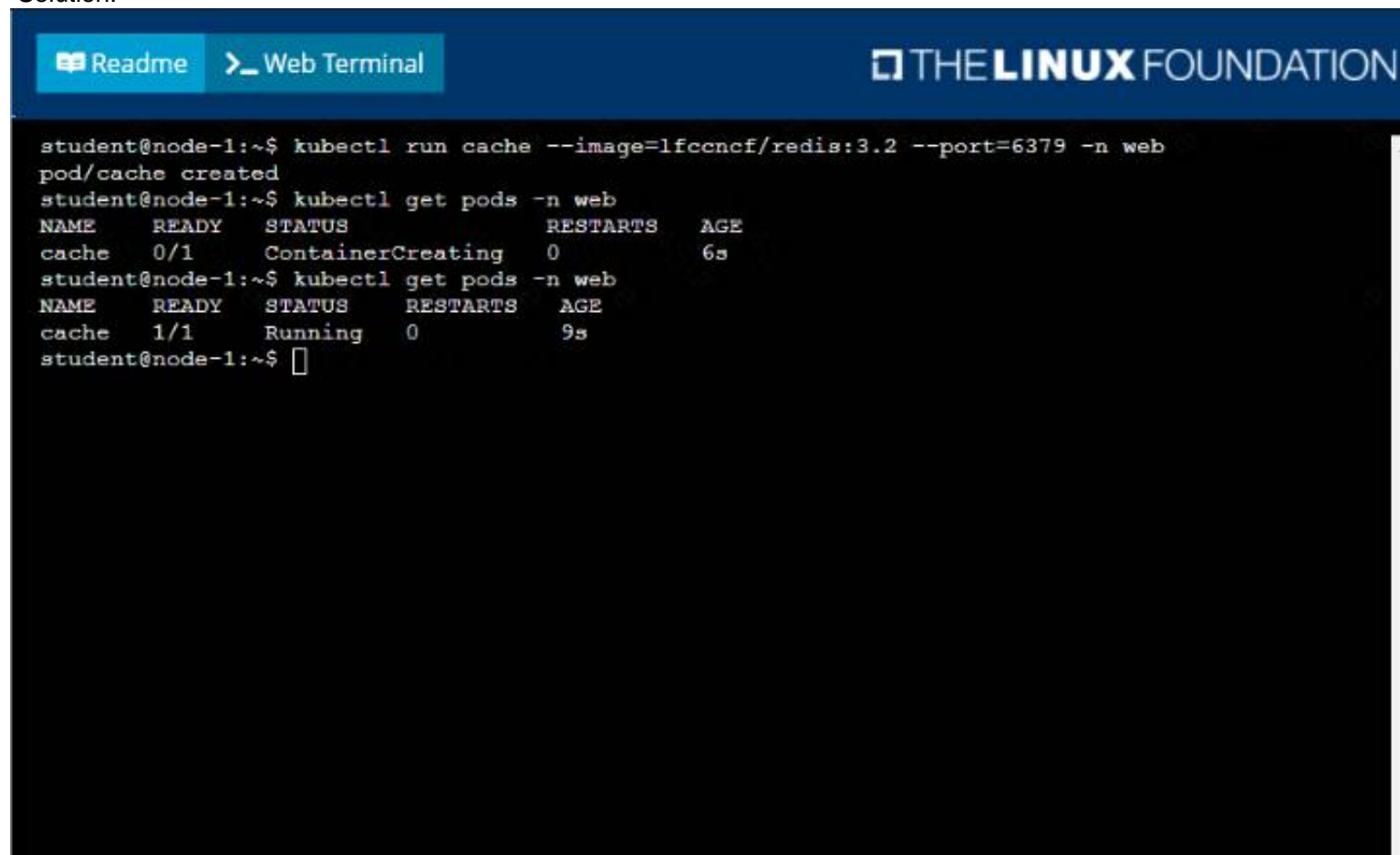
Update the app-adeployment in theproductionnamespace to run as therestrictedserviceservice account. The service account has already been created.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



```
student@node-1:~$ kubectl run cache --image=lfcncf/redis:3.2 --port=6379 -n web
pod/cache created
student@node-1:~$ kubectl get pods -n web
NAME      READY   STATUS             RESTARTS   AGE
cache     0/1     ContainerCreating   0           6s
student@node-1:~$ kubectl get pods -n web
NAME      READY   STATUS    RESTARTS   AGE
cache     1/1     Running   0           9s
student@node-1:~$
```

NEW QUESTION 7

Exhibit:



Context

Developers occasionally need to submit pods that run periodically. Task

Follow the steps below to create a pod that will start at a predetermined time and]which runs to completion

only once each time it is started:

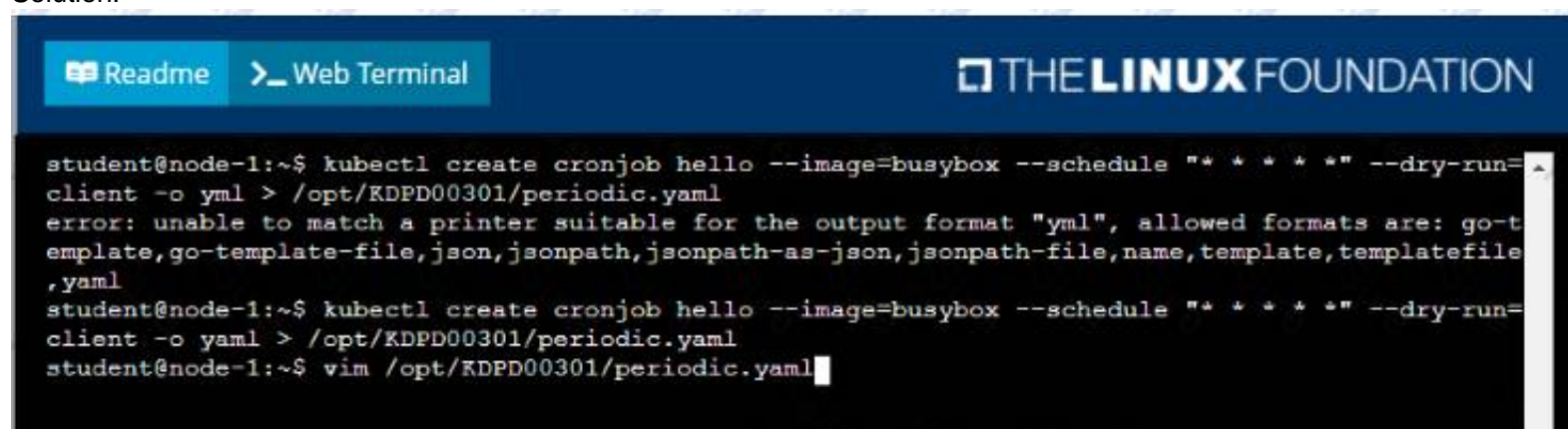
- Create a YAML formatted Kubernetes manifest /opt/KDPD00301/periodic.yaml that runs the following shell command: date in a single busybox container. The command should run every minute and must complete within22seconds or be terminated by Kubernetes. The Cronjob namp and container name should both be hello
- Create the resource in the above manifest and verify that the job executes successfully at least once

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



```
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "*" * * * * --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
```


Readme
Web Terminal

THE **LINUX** FOUNDATION

```

apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
              restartPolicy: Never
  schedule: '*/* * * * *'
  startingDeadlineSeconds: 22
  concurrencyPolicy: Allow

```

19,26 All

Readme
Web Terminal

THE **LINUX** FOUNDATION

```

student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob
NAME      SCHEDULE      SUSPEND   ACTIVE   LAST SCHEDULE   AGE
hello     */1 * * * *   False    0        <none>          6s
student@node-1:~$

```

NEW QUESTION 8

Exhibit:



Context

As a Kubernetes application developer you will often find yourself needing to update a running application. Task Please complete the following:

- Update the app deployment in the kdpd00202 namespace with a maxSurge of 5% and a maxUnavailable of 2%
- Perform a rolling update of the web1 deployment, changing the lfcncf/ngmx image version to 1.13
- Roll back the app deployment to the previous version

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

Readme
Web Terminal

THE **LINUX** FOUNDATION

```

student@node-1:~$ kubectl edit deployment app -n kdpd00202

```


Readme
Web Terminal

THE LINUX FOUNDATION

```
uid: 1dfa2527-5c61-46a9-8dd3-e24643d3ce14
spec:
  progressDeadlineSeconds: 600
  replicas: 10
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 5%
      maxUnavailable: 2
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
    spec:
      containers:
      - image: lfocncf/nginx:1.13
        imagePullPolicy: IfNotPresent
        name: nginx
        ports:
        - containerPort: 80
          protocol: TCP
:wq!
```

Readme
Web Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl edit deployment app -n kdpd00202
deployment.apps/app edited
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 8 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 8 of 10 updated replicas are available...
Waiting for deployment "app" rollout to finish: 9 of 10 updated replicas are available...
deployment "app" successfully rolled out
student@node-1:~$ kubectl rollout undo deployment app -n kdpd00202
deployment.apps/app rolled back
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 7 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 9 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "app" rollout to finish: 8 of 10 updated replicas are available...
Waiting for deployment "app" rollout to finish: 9 of 10 updated replicas are available...
deployment "app" successfully rolled out
student@node-1:~$
```

NEW QUESTION 10

.....

Relate Links

100% Pass Your CKAD Exam with ExamBible Prep Materials

<https://www.exambible.com/CKAD-exam/>

Contact us

We are proud of our high-quality customer service, which serves you around the clock 24/7.

Viste - <https://www.exambible.com/>