Linux Foundation CKAD인증시험

참고: Pass4Test에서 Google Drive로 공유하는 무료, 최신 CKAD 시험 문제집이 있습니다: https://drive.google.com/open?id=1aqntBYdx_7t-UACmpiPdDDoRCtPygfsj

Linux Foundation CKAD인증덤프가 Pass4Test전문가들의 끈임 없는 노력 하에 최고의 버전으로 출시되었습니다. 여러분의 꿈을 이루어드리려고 말이죠. IT업계에서 자기만의 자리를 잡고 싶다면Linux Foundation CKAD인증시험이 아주 좋은 자격증입니다. 만약Linux Foundation CKAD인증시험 자격증이 있다면 일에서도 많은 변화가 있을 것입니다, 연봉상승은 물론, 자기자신만의 공간도 넓어집니다.

리눅스 재단 CKAD 자격증 시험은 Kubernetes 애플리케이션 개발 능력을 증명하고자 하는 개발자들을 위한 귀중한 자격증 프로그램입니다. 이 시험은 실제 상황에서의 성능을 기반으로 한 실무 시험으로, 후보자의 Kubernetes 애플리케이션 배포, 구성 및 관리 능력을 평가합니다. CKAD 자격증은 전 세계적으로 인정받으며, 개발자들의 경력 전망을 향상시키고 취업 기회를 늘리는 데 도움이 될 수 있습니다.

Linux Foundation Certified Kubernetes Application Developer (CKAD) 인증 시험은 개발자에게 Kubernetes 응용 프로그램 개발에 대한 기술과 전문 지식을 보여줄 수있는 기회를 제공하는 인기있는 시험입니다. 이 시험은 매일 Kubernetes와 함께 일하는 개발자의 기술을 평가하고 자신의 기술과 지식을 검증하려고합니다.

Linux Foundation CKAD (Certified Kubernetes 응용 프로그램 개발자) 인증 시험은 Kubernetes 애플리케이션 개발에 능숙성을 보여 주려는 개인에게 인기있는 인증 시험입니다. Kubernetes는 컨테이너화 된 응용 프로그램의 배포, 스케일링 및 관리에 사용되는 오픈 소스 시스템입니다. Kubernetes가 인기를 얻음에 따라 CKAD 인증을받은 전문가에 대한 수요는 급격히 증가하고 있습니다. 인증 시험은 후보자의 Kubernetes 응용 프로그램을 설계, 빌드, 구성 및 노출시키는 능력을 테스트하도록 설계되었습니다.

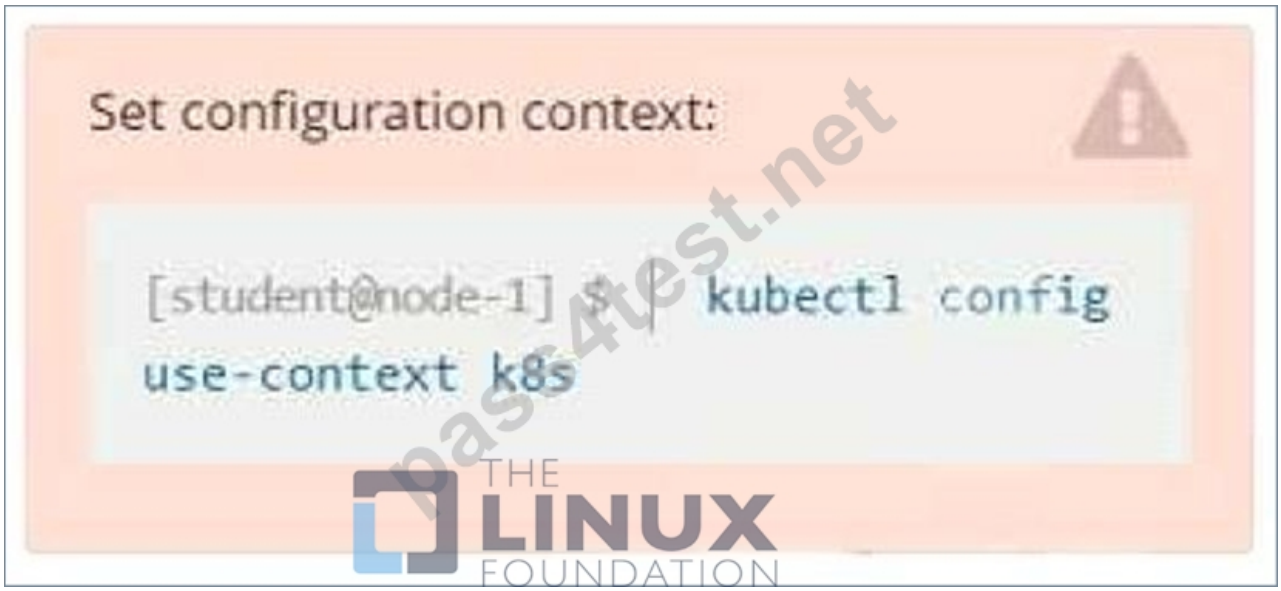
>> CKAD퍼펙트 최신 덤프공부 <<

최근 인기시험 CKAD퍼펙트 최신 덤프공부 덤프샘플문제

우리Pass4Test 사이트에Linux Foundation CKAD관련자료의 일부 문제와 답 등 문제들을 제공함으로 여러분은 무료로 다운받아 체험해보실 수 있습니다. 여러분은 이것이야 말로 알맞춤이고, 전면적인 여러분이 지금까지 갖고 싶었던 문제집이라는 것을 느끼게 됩니다.

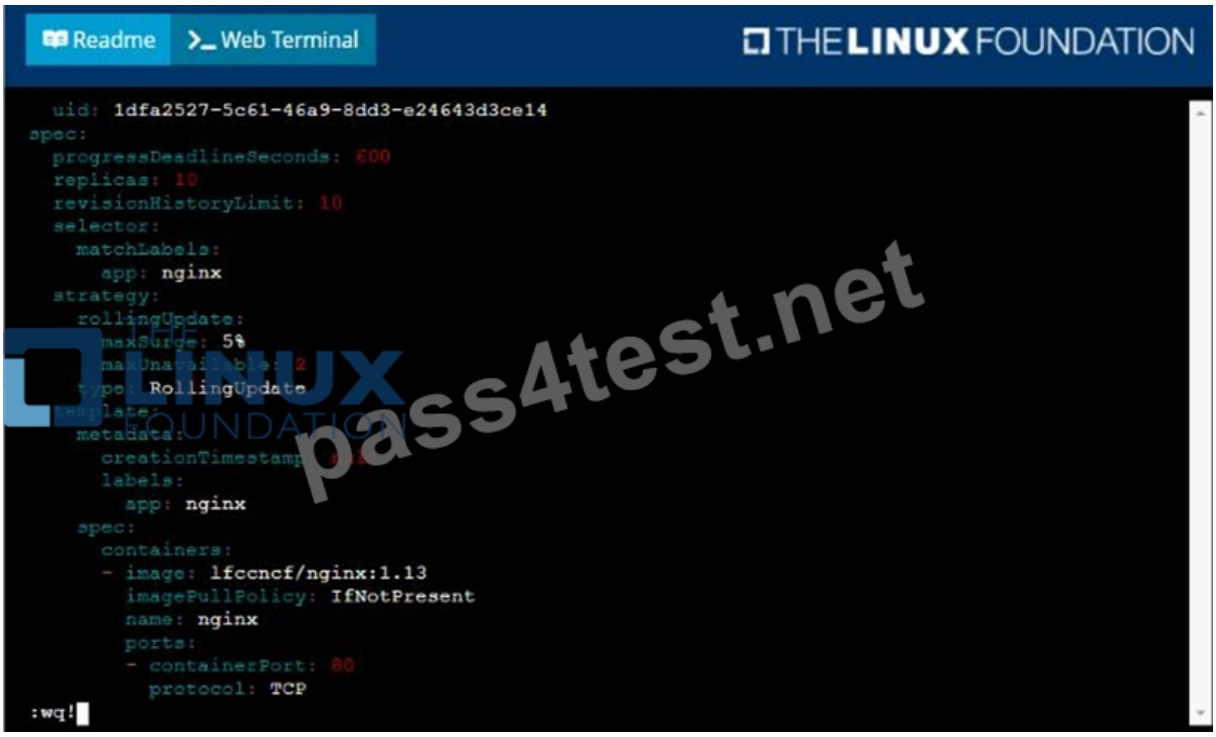
최신 Kubernetes Application Developer CKAD 무료샘플문제 (Q172-Q177):

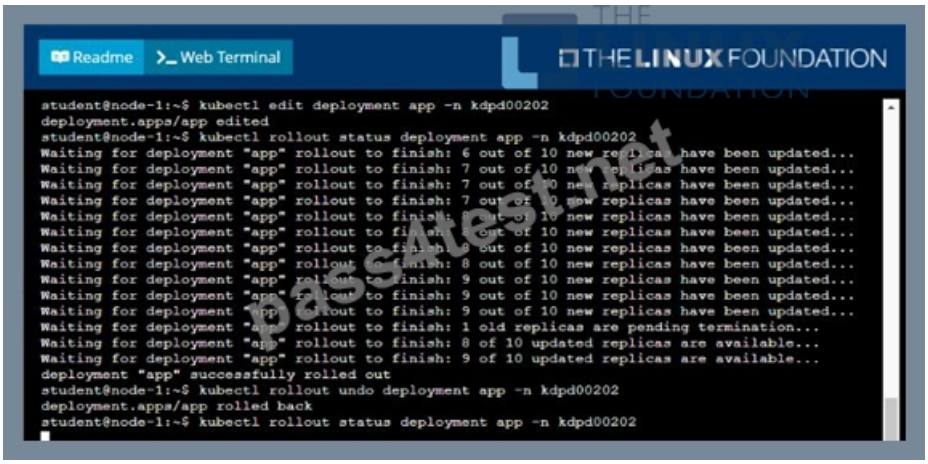
질문 # 172
Exhibit:

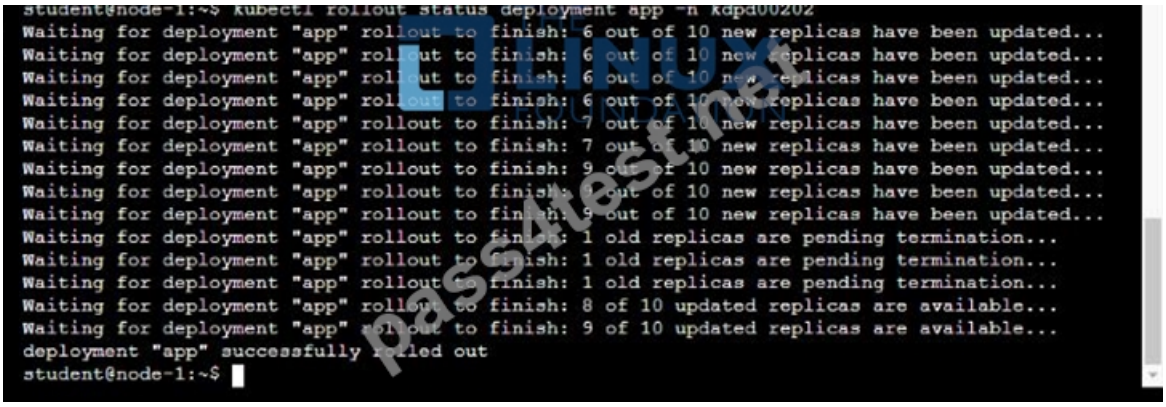


Context
As a Kubernetes application developer you will often find yourself needing to update a running application.
Task
Please complete the following:
* Update the app deployment in the kdpd00202 namespace with a maxSurge of 5% and a maxUnavailable of 2%
* Perform a rolling update of the web1 deployment, changing the lfccncf/ngmx image version to 1.13
* Roll back the app deployment to the previous version

- A. Solution:

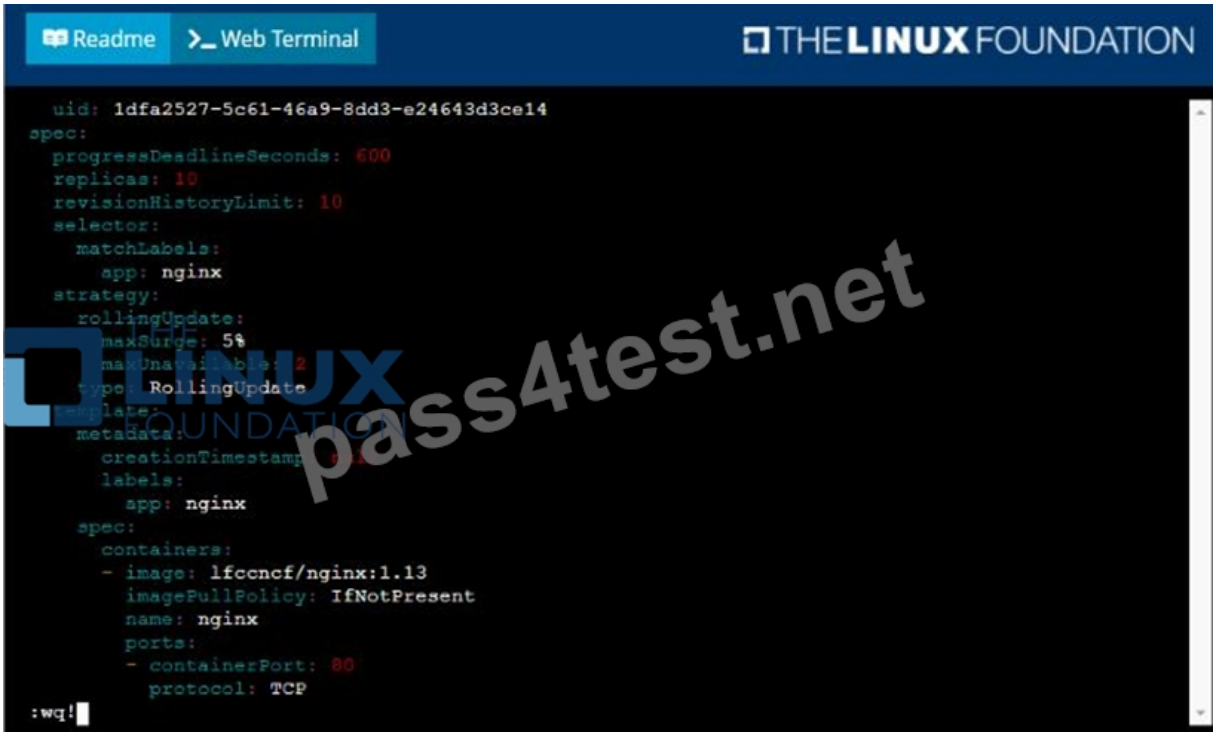


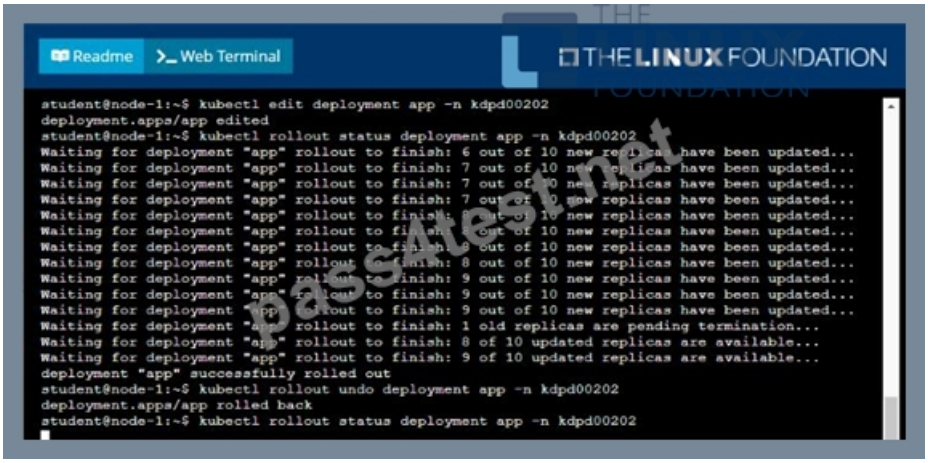

Readme
Web Terminal
THE LINUX FOUNDATION
student@node-1:~$ kubectl edit deployment app -n kdpd00202
deployment.apps/app edited
student@node-1:~$ kubectl rollout status deployment app -n kdpd00202
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for 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: 9 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

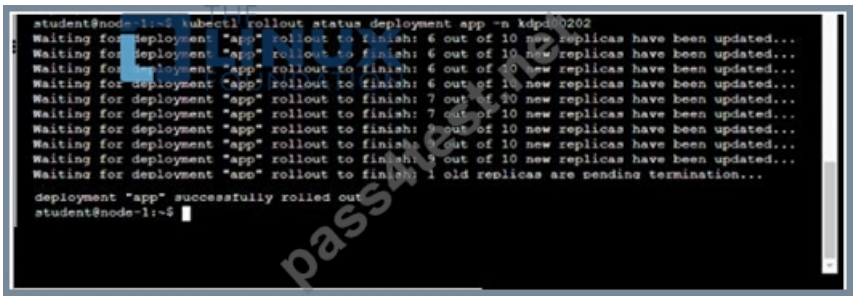

Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "app" rollout to finish: 6 out of 10 new replicas have been updated...
Waiting for deployment "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: 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:~$

- B. 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:
labels:
app: nginx
spec:
containers:
- image: lfccncf/nginx:1.13
imagePullPolicy: IfNotPresent
name: nginx
ports:
- containerPort: 80
protocol: TCP
:wq!





정답：A

질문 # 173
You are building a microservices application on Kubernetes, where two services, and 'service-b' , need to communicate with each other securely. 'Service-b' needs to expose a secure endpoint that is only accessible by 'service-a'. Describe how you would implement this using Kubernetes resources, including the configuration for the 'service-b' endpoint.

정답：

설명：
See the solution below with Step by Step Explanation.
Explanation:
Solution (Step by Step) :
1. Define a Kubernetes Secret:
- Create a Kubernetes secret to store the certificate and key pair for 'service-W. This secret will be used to secure the communication.
- Example:

```
apiVersion: v1
kind: Secret
metadata:
  name: service-b-tls
type: kubernetes.io/tls
data:
  tls.crt:
  tls.key:
```

2. Configure 'service-b' Deployment: - Define a Deployment for 'service-b' , specifying a container that uses the secret for TLS. - Ensure that the container has the required dependencies and configuration to use TLS. - Example:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: service-b-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: service-b
  template:
    metadata:
      labels:
        app: service-b
    spec:
      containers:
      - name: service-b
        image: your-image:latest
        ports:
          - containerPort: 8443
        volumeMounts:
          - name: service-b-tls
            mountPath: /var/tls/
      volumes:
        - name: service-b-tls
          secret:
            secretName: service-b-tls
```


3. Define a Kubernetes Service for 'service-b'.' - Create a Service for 'service-b' that exposes the secure endpoint on a specific port (e.g., 8443) and uses the LoadBalancer' type for external access. - Use the 'targetPort' field to specify the container port that 'service-b' is listening on. - Example:

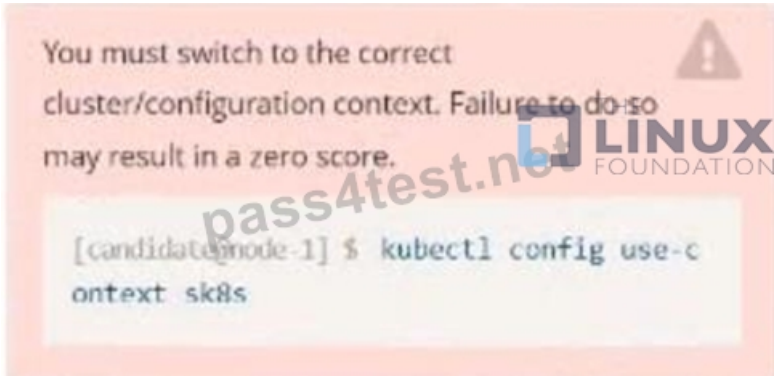
```yaml
apiVersion: v1
kind: Service
metadata:
  name: service-b-service
spec:
  type: LoadBalancer
  ports:
    - protocol: TCP
      port: 8443
      targetPort: 8443
  selector:
    app: service-b
```


4. Configure 'service-a' Deployment: - Define a Deployment for 'service-a', specifying a container that uses the secret for TLS when connecting to service-W. - Example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: service-a-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: service-a
  template:
    metadata:
      labels:
        app: service-a
    spec:
      containers:
      - name: service-a
        image: your-image:latest
        ports:
          - containerPort: 8080
        volumeMounts:
            - name: service-b-tls
              mountPath: /var/tls/
        volumes:
          - name: service-b-tls
            secret:
              secretName: service-b-tls
```


5. Update 'service-a' Container Configuration: - Within the 'service-a' container, ensure the application is configured to use the certificate and key from the mounted volume ('Ivar/tls/') for secure communication with 'service-b'. 6. Verify Secure Communication: - Use 'kubectl get pods' to check the status of both 'service-a' and 'service-W pods. - Test the communication between 'service-a' and 'service-b' by sending requests from the 'service-a' pod to the secure endpoint of 'service-b'. - Verify that the communication is secure and that 'service-a' can successfully access the endpoint. Notes: - You may need to adjust the port numbers and image names in the examples to match your specific setup. - Make sure you have the certificate and key in the correct format and base64 encoded before creating the Secret. - You can also use other methods like a Service Account and Role-Based Access Control (RBAC) to restrict access to the secure endpoint, if needed. - This is a simplified example and additional security measures may be required based on your application's requirements. ,

질문 # 174

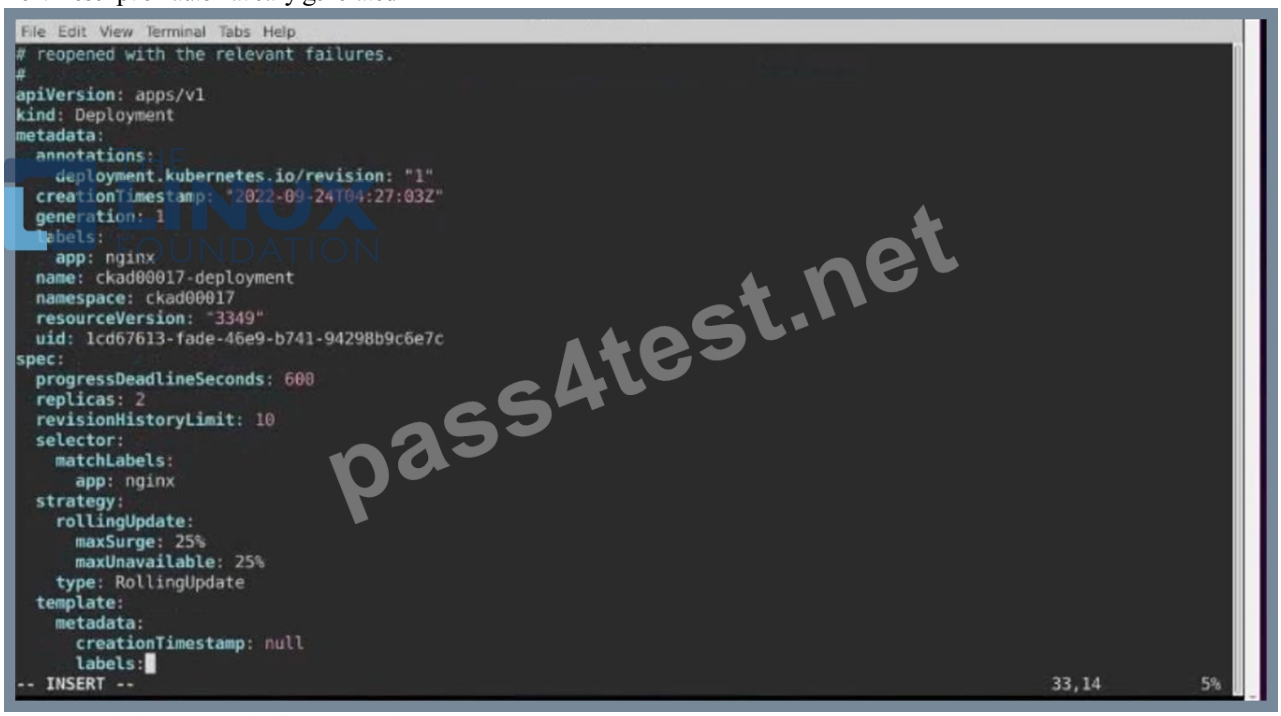


Task:
1) First update the Deployment cka00017-deployment in the ckad00017 namespace:
Role userUI

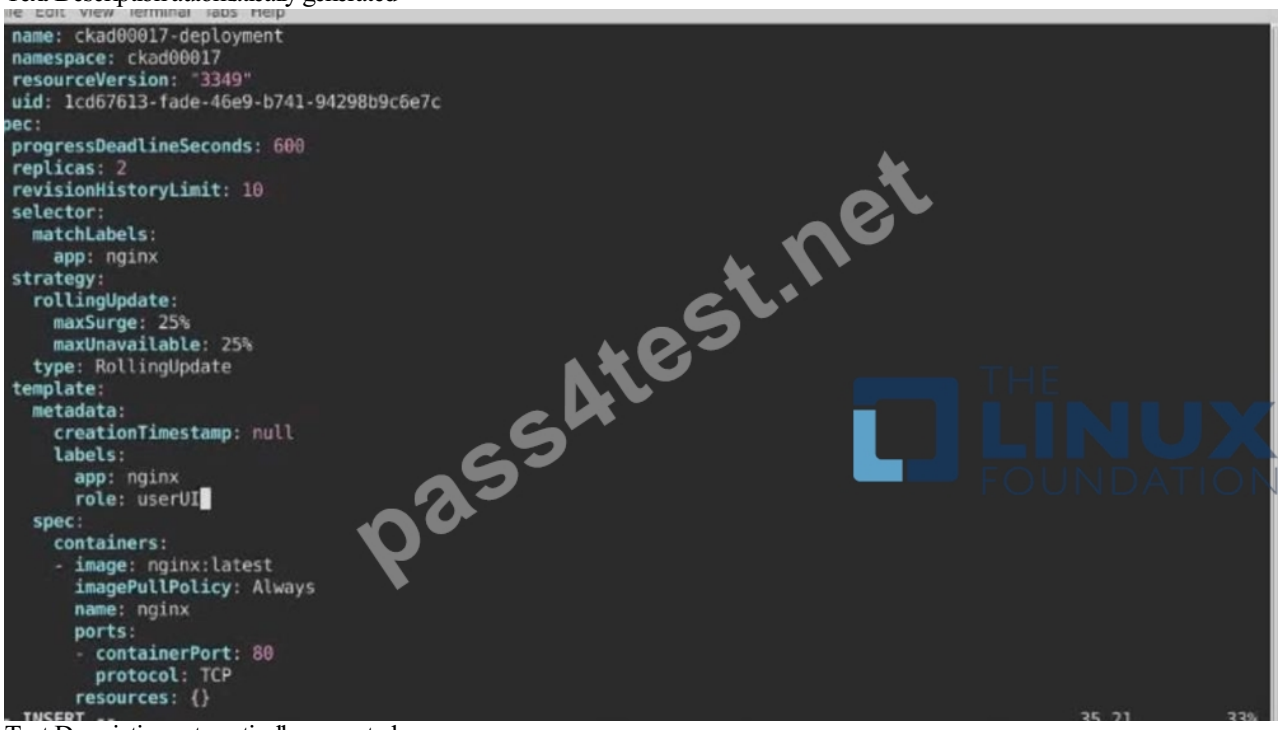
2) Next, Create a NodePort Service named cherry in the ckad00017 nmespace exposing the ckad00017-deployment Deployment on TCP port 8888 See the solution below.

정답：

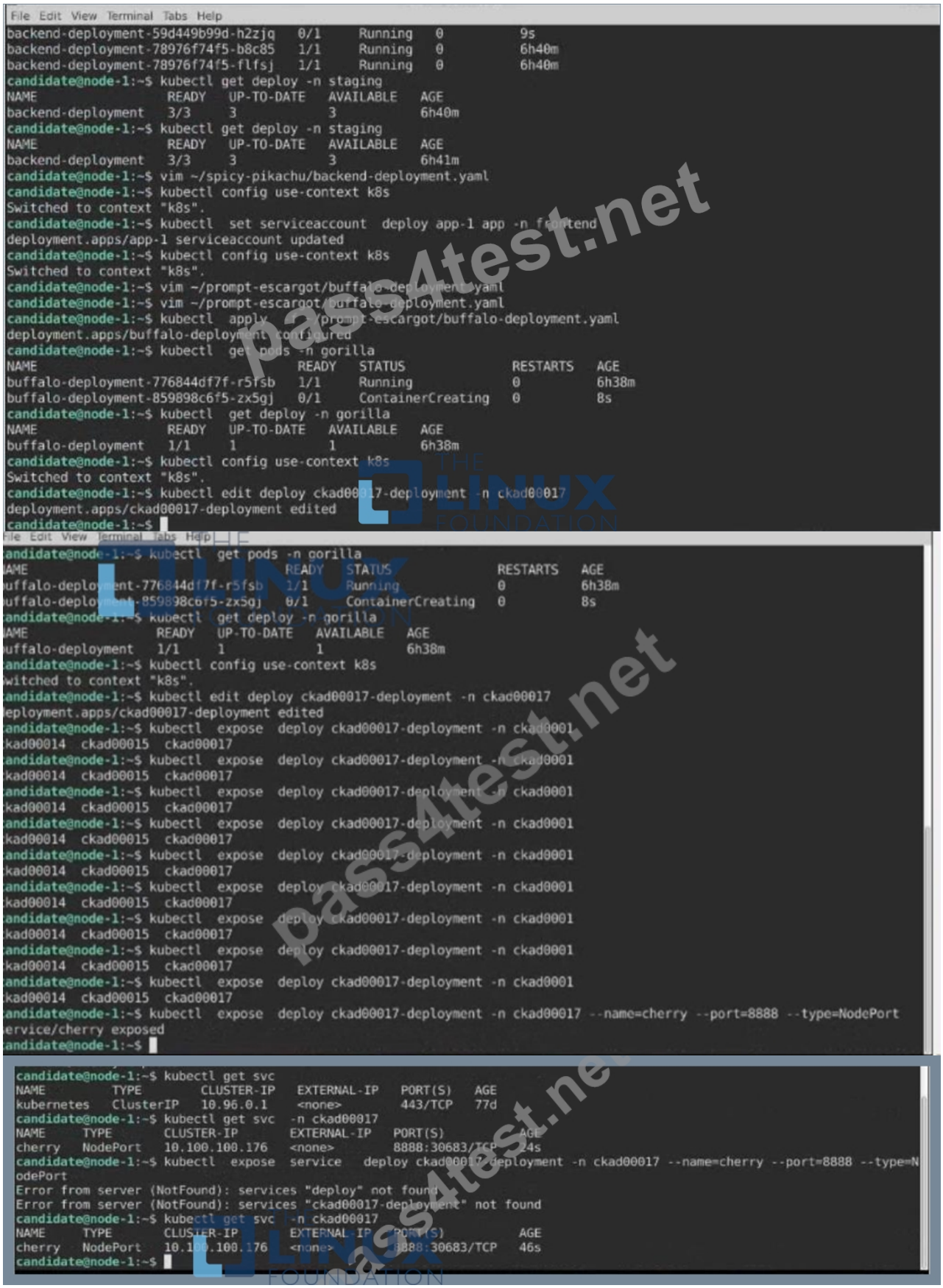
설명：
Explanation
Solution:
Text Description automatically generated



Text Description automatically generated



Text Description automatically generated



```
candidate@node-1:~$ kubectl  expose  service   deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=N
odePort
Error from server (NotFound): services "deploy" not found
Error from server (NotFound): services "ckad00017-deployment" not found
candidate@node-1:~$ kubectl get svc  -n ckad00017
NAME     TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
cherry   NodePort   10.100.100.176   <none>        8888:30683/TCP   46s
candidate@node-1:~$ history
    1  vi ~/spicy-pikachu/backend-deployment.yaml
    2  kubectl config use-context sk8s
    3  vim .vimrc
    4  vim ~/spicy-pikachu/backend-deployment.yaml
    5  kubectl  apply  -f ~/spicy-pikachu/backend-deployment.yaml
    6  kubectl get  pods -n staging
    7  kubectl get deploy -n staging
    8  vim ~/spicy-pikachu/backend-deployment.yaml
    9  kubectl config use-context k8s
   10  kubectl  set serviceaccount  deploy app-1 app -n frontend
   11  kubectl config use-context k8s
   12  vim ~/prompt-escargot/buffalo-deployment.yaml
   13  kubectl  apply  -f ~/prompt-escargot/buffalo-deployment.yaml
   14  kubectl  get pods -n gorilla
   15  kubectl  get deploy -n gorilla
   16  kubectl config use-context k8s
   17  kubectl edit deploy ckad00017-deployment -n ckad00017
   18  kubectl  expose  deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
   19  kubectl get svc
   20  kubectl get svc  -n ckad00017
   21  kubectl  expose  service   deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
   22  kubectl get svc  -n ckad00017
   23  history
candidate@node-1:~$
```

질문 # 175
You are developing a new microservice that requires access to a database deployed in a different namespace. You want to configure a ServiceAccount and RoleBinding to provide the necessary permissions for the microservice to connect to the database.

정답：

설명：
See the solution below with Step by Step Explanation.
Explanation:
Solution (Step by Step) :
1. Create a ServiceAccount:
- Create a ServiceAccount in the namespace where our microservice is deployed:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-microservice-sa
  namespace: my-microservice-namespace
```

2. Create a Role: - Create a Role in the namespace where the database is deployed, granting access to the database resources:

```
ind: Role
etadata:
  name: my-database-access-role
  namespace: my-database-namespace
ules:
  apiGroups: [""]
  resources: ["pods", "services", "secrets"]
  verbs: ["get", "list", "watch"]
  apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["get", "list", "watch", "create", "update", "delete"]
  apiGroups: ["batch"]
  resources: ["jobs", "cronjobs"]
  verbs: ["get", "list", "watch", "create", "update", "delete"]
```

3. Create a ROIeBinding: - Create a RoleBinding in the database namespace to bind the Role to the ServiceAccount:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: my-database-access-rolebinding
  namespace: my-database-namespace
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: my-database-access-role
subjects:
- kind: ServiceAccount
  name: my-microservice-sa
  namespace: my-microservice-namespace
```

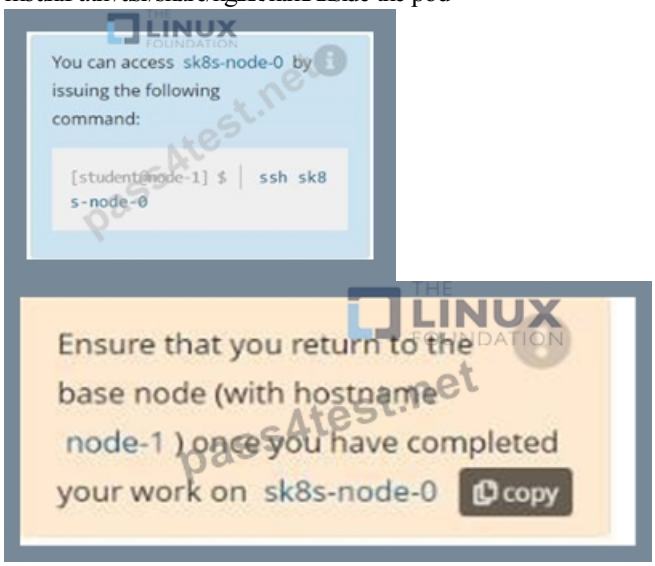
4. Apply the Configuration: - Apply the created ServiceAccount, Role, and Roledinding using 'kubectl apply -r commands: bash kubectl apply -f my-microservice-sa_yaml kubectl apply -f my-database-access-role-yaml kubectl apply -f my-database-access-rolebinding.yaml 5. Configure the Microservice: - Mount the ServiceAccount token as a secret within the microservice's pod:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-microservice
  namespace: my-microservice-namespace
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-microservice
  template:
    metadata:
      labels:
        app: my-microservice
    spec:
      serviceAccountName: my-microservice-sa
      containers:
      - name: my-microservice
        image: my-microservice-image:latest
        volumeMounts:
        - name: sa-token
          mountPath: /var/run/secrets/kubernetes.io/serviceaccount
      volumes:
      - name: sa-token
        secret:
          secretName: default-token-my-microservice-sa
```

6. Verify Permissions: - Access the database from the microservice pod to verify that the required permissions are granted.

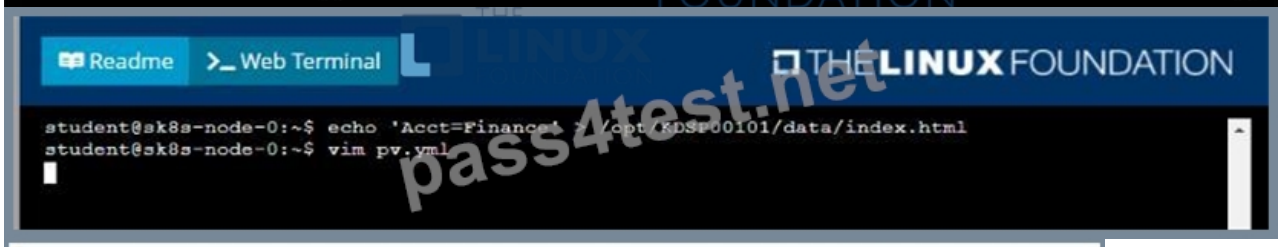
질문 # 176

Context
A project that you are working on has a requirement for persistent data to be available.
Task
To facilitate this, perform the following tasks:
* Create a file on node sk8s-node-0 at /opt/KDSP00101/data/index.html with the content Acct=Finance
* Create a PersistentVolume named task-pv-volume using hostPath and allocate 1Gi to it, specifying that the volume is at /opt/KDSP00101/data on the cluster's node. The configuration should specify the access mode of ReadWriteOnce . It should define the StorageClass name exam for the PersistentVolume , which will be used to bind PersistentVolumeClaim requests to this PersistenetVolume.
* Create a PefsissentVolumeClaim named task-pv-claim that requests a volume of at least 100Mi and specifies an access mode of ReadWriteOnce
* Create a pod that uses the PersistentVolmeClaim as a volume with a label app: my-storage-app mounting the resulting volume to a mountPath /usr/share/nginx/html inside the pod



정답：

설명：
See the solution below.
Explanation
Solution:



Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
spec:
  capacity:
    storage: 1Gi
  accessModes:
  - ReadWriteOnce
  storageClassName: storage
  hostPath:
    path: /opt/KDSP00101/data
    type: Directory
```

Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: storage
```

```
student@sk8s-node-0:~$ kubectl create -f pv.yml
persistentvolume/task-pv-volume created
student@sk8s-node-0:~$ kubectl create -f pvc.yml
persistentvolumeclaim/task-pv-claim created
student@sk8s-node-0:~$ kubectl get pv
NAME             CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                   STO
RAGECLASS   REASON   AGE
task-pv-volume   1Gi        RWO            Retain           Bound    default/task-pv-claim   sto
rage                11s
student@sk8s-node-0:~$ kubectl get pvc
NAME            STATUS   VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim   Bound    task-pv-volume   1Gi        RWO            storage        9s
student@sk8s-node-0:~$ vim pod.yml
```

Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod
  labels:
    app: my-storage-app
spec:
  containers:
  - name: myfrontend
    image: nginx
    volumeMounts:
    - mountPath: "/usr/share/nginx/html"
      name: mypod
  volumes:
    - name: mypod
      persistentVolumeClaim:
        claimName: task-pv-claim
~
~
~
~
~
~
~
~
~
                                                    17,32         All
```

```
student@sk8s-node-0:~$ kubectl create -f pod.yml
pod/mypod created
student@sk8s-node-0:~$ kubectl get
```

질문 # 177
.....

IT자격증을 많이 취득하여 IT업계에서 자신만의 단단한 자리를 보장하는것이 여러분들의 로망이 아닐가 싶습니다. Pass4Test의 완벽한 Linux Foundation인증 CKAD덤프는 IT전문가들이 자신만의 노하우와 경험으로 실제Linux Foundation인증 CKAD시험문제에 대비하여 연구제작한 완벽한 작품으로서 100%시험통과율을 보장합니다.

CKAD최고품질 덤프데모 다운로드: https://www.pass4test.net/CKAD.html

- CKAD최신 인증시험정보 □ CKAD인증덤프 샘플문제 □ CKAD인기시험 □ 시험 자료를 무료로 다운로드하려면➥ kr.fast2test.com □을 통해【 CKAD 】를 검색하십시오CKAD최신 인증시험정보
- CKAD자격증공부 □ CKAD시험패스 가능한 공부문제 □ CKAD시험패스 인증공부자료 □ 지금▸ www.itdumpskr.com ◂에서□ CKAD □를 검색하고 무료로 다운로드하세요CKAD덤프최신문제
- CKAD시험응시료 □ CKAD최신 인증시험 기출자료 □ CKAD자격증공부 □ 무료 다운로드를 위해 지금➡ www.itdumpskr.com □□□에서[CKAD]검색CKAD덤프최신문제
- 시험패스 가능한 CKAD퍼펙트 최신 덤프공부 덤프 샘플문제 다운받기 □ ▸ www.itdumpskr.com ◂의 무료 다운로드➠ CKAD □페이지가 지금 열립니다CKAD덤프최신문제
- 시험패스 가능한 CKAD퍼펙트 최신 덤프공부 인증덤프자료 □ “ www.pass4test.net ”을 통해 쉽게✔ CKAD □✔□무료 다운로드 받기CKAD높은 통과율 시험자료
- 시험패스 가능한 CKAD퍼펙트 최신 덤프공부 인증덤프자료 □ [www.itdumpskr.com]에서[CKAD]를 검색하고 무료로 다운로드하세요CKAD덤프최신문제
- 시험패스 가능한 CKAD퍼펙트 최신 덤프공부 인증덤프자료 □ 지금➡ www.exampassdump.com □□□에서➡ CKAD □□□를 검색하고 무료로 다운로드하세요CKAD최신 인증시험 기출자료
- CKAD퍼펙트 최신 덤프공부 완벽한 시험자료 □ ➥ www.itdumpskr.com □은➽ CKAD □무료 다운로드를 받을 수 있는 최고의 사이트입니다CKAD높은 통과율 시험자료
- CKAD퍼펙트 최신 덤프공부 완벽한 시험자료 □ 검색만 하면“ www.exampassdump.com ”에서➡ CKAD □□□무료 다운로드CKAD높은 통과율 시험자료
- CKAD퍼펙트 최신 덤프공부 덤프로 시험정복하기 □ □ www.itdumpskr.com □을(를) 열고▹ CKAD ◃를 입력하고 무료 다운로드를 받으십시오CKAD퍼펙트 덤프데모문제 보기
- CKAD퍼펙트 덤프데모문제 보기 □ CKAD시험패스 가능한 공부문제 □ CKAD인증시험 인기 덤프자료 □ □ ➽ www.pass4test.net □에서☀ CKAD □☀□를 검색하고 무료 다운로드 받기CKAD최고덤프
- myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, shortcourses.russellcollege.edu.au, www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, blogfreely.net, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, Disposable vapes

Pass4Test CKAD 최신 PDF 버전 시험 문제집을 무료로 Google Drive에서 다운로드하세요: https://drive.google.com/open?id=1aqntBYdx_7t-UACmpiPdDDoRCtPygfsj