2022年CKA认证题库

2022年CKA认证题库

考试说明:

1. 考试前考官会通过考试系统的共享桌面检查你电脑上的后台进程，通过摄像头确认核实身份信息，至少需要身份证+护照或者信用卡两个证件以及检查你考试周边的环境。由于考试官网在美国，服务器也在美国，导致国内无法打开或者访问系统较慢，因此建议要具备kexue上网条件。考试时浏览器只允许打开两个窗口，一个是考试系统，另外一个可以查阅k8s官方文档，所以建议提前把题库里的每道考题官网链接添加到浏览器书签里，方便快速查找。

2. 更多信息请看中国区官网: https://training.linuxfoundation.cn/help
3. 编辑/拷贝YAML文件的时候，建议使用paste模式，解决粘贴乱序问题。（见文档末尾附录）

题库说明:

1. 题库为CKA认证原题题库，每月都会有朋友参加考试，保证题库的准确性。题库永久更新，直到考过为止。最近配置了一套CKA认证的模拟系统，按照考试系统搭建的，已集成题库，可以直接实操练习做题，让你省心省事省力，快速有效的考取CKA认证，需要的朋友可以在淘宝店铺里搜索。

操作系统版本	kubernetes版本
Ubuntu 18.04	1.22

2. 根据店主刷题以及考过的朋友考试经分享，近一年多17道题的考点基本没变，只是改变了题目顺序和部分参数，考试就是题库里的这17道题，熟练练习考过没什么难度。

更新说明：

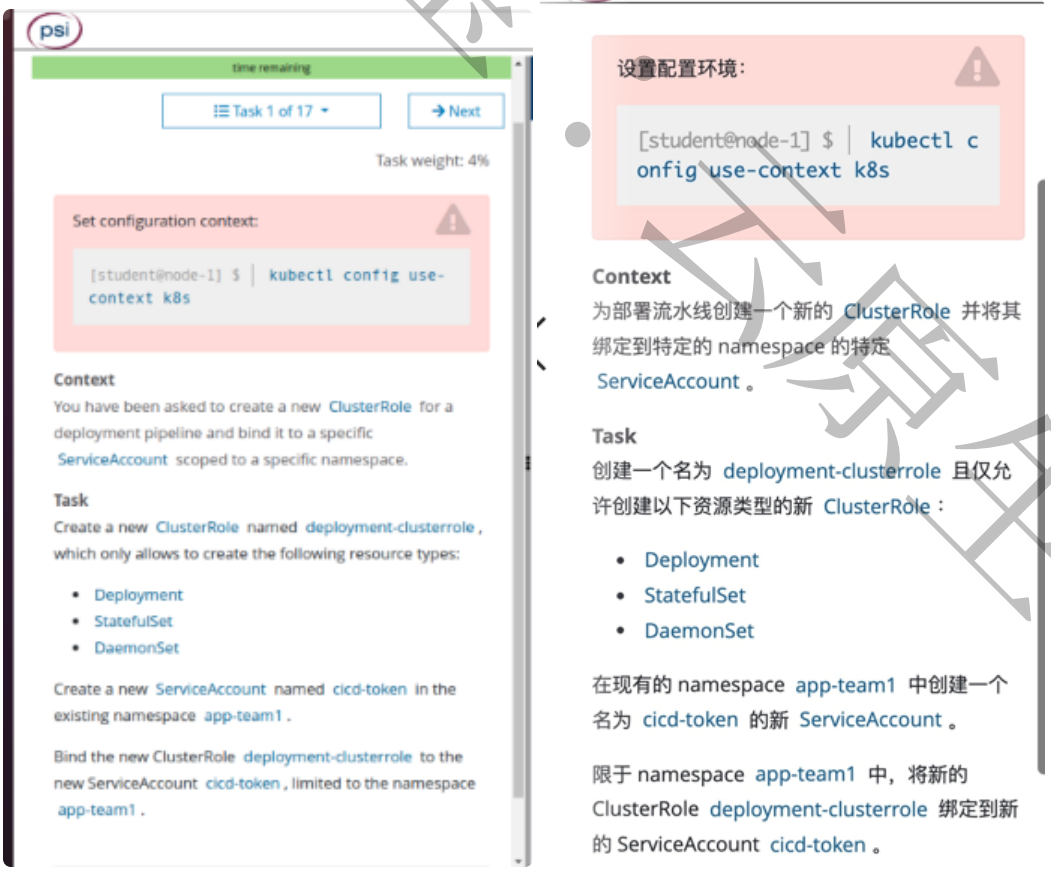
2021年12月：

1. 第二题 设置节点不可用参数变更：--delete-emptydir-data；
2. 第三题 k8s升级版本为从1.22.1到1.22.2；
3. 第四题 新增etcd还原新方法；

2022年1月：

1. 考题无变化；

第一题：权限控制RBAC



考题概述：

创建名称 deployment-clusterrole 的 ClusterRole，该角色具备创建 Deployment、Statefulset、Daemonset 的权限，在命名空间 app-team1 中创建名称为 cicd-token 的 ServiceAccount，绑定 ClusterRole 到 ServiceAccount，且限定命名空间为 app-team1。

考题解析：

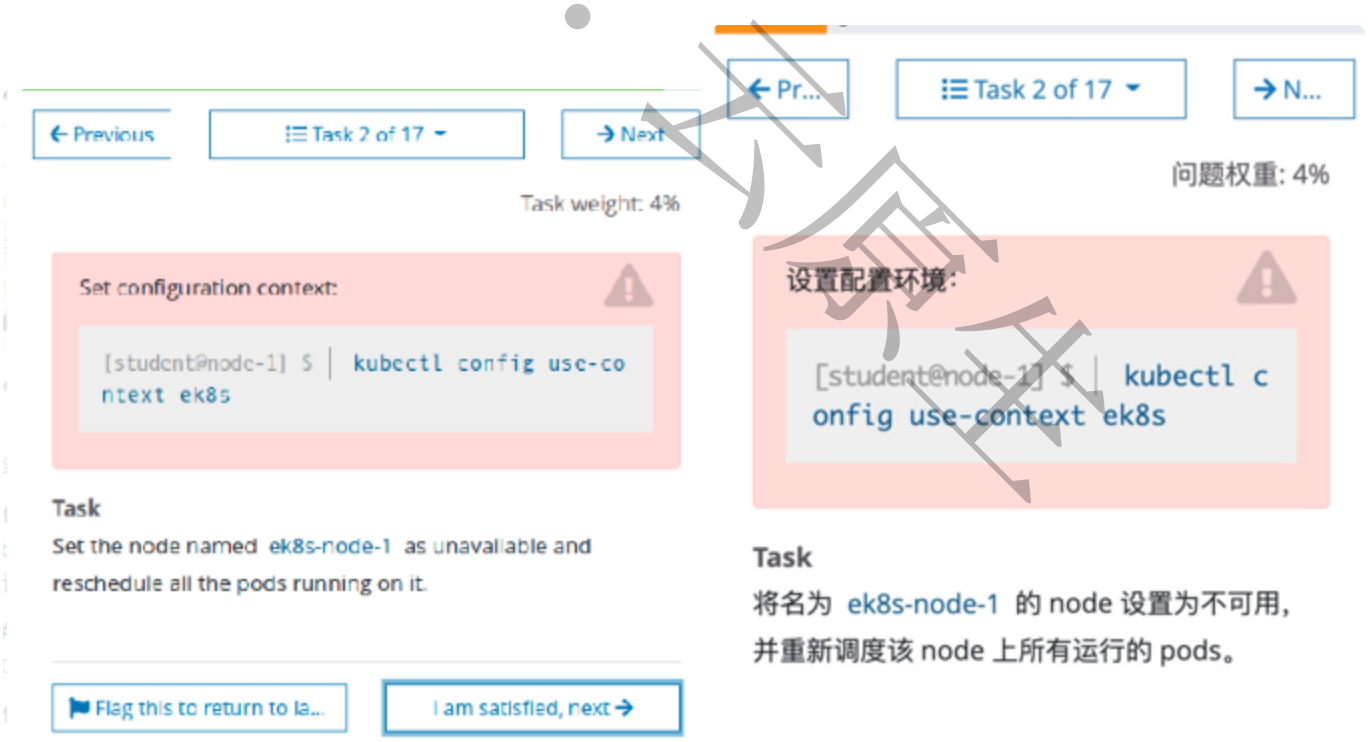
需要熟悉创建 serviceaccount、clusterrole 和 rolebinding 的方法，需要限定在 ns 级别，因此最好使用 rolebinding

参考方法：

```
Shell | 复制代码
1  切换 context
2  kubectl create ns app-team1 （题库练习执行，命名空间考试系统已存在）
3  kubectl create serviceaccount cicd-token -n app-team1
4  kubectl create clusterrole deployment-clusterrole --verb=create --
   resource=deployments,statefulsets,daemonsets
5  kubectl -n app-team1 create rolebinding cicd-clusterrole --
   clusterrole=deployment-clusterrole --serviceaccount=app-team1:cicd-token
```

考点官网链接：https://kubernetes.io/docs/reference/access-authn-authz/rbac/

第二题：设置节点不可用



考题概述：

设置 ek8s-node-1 节点为不可用、重新调度该节点上的所有 pod

考题解析：

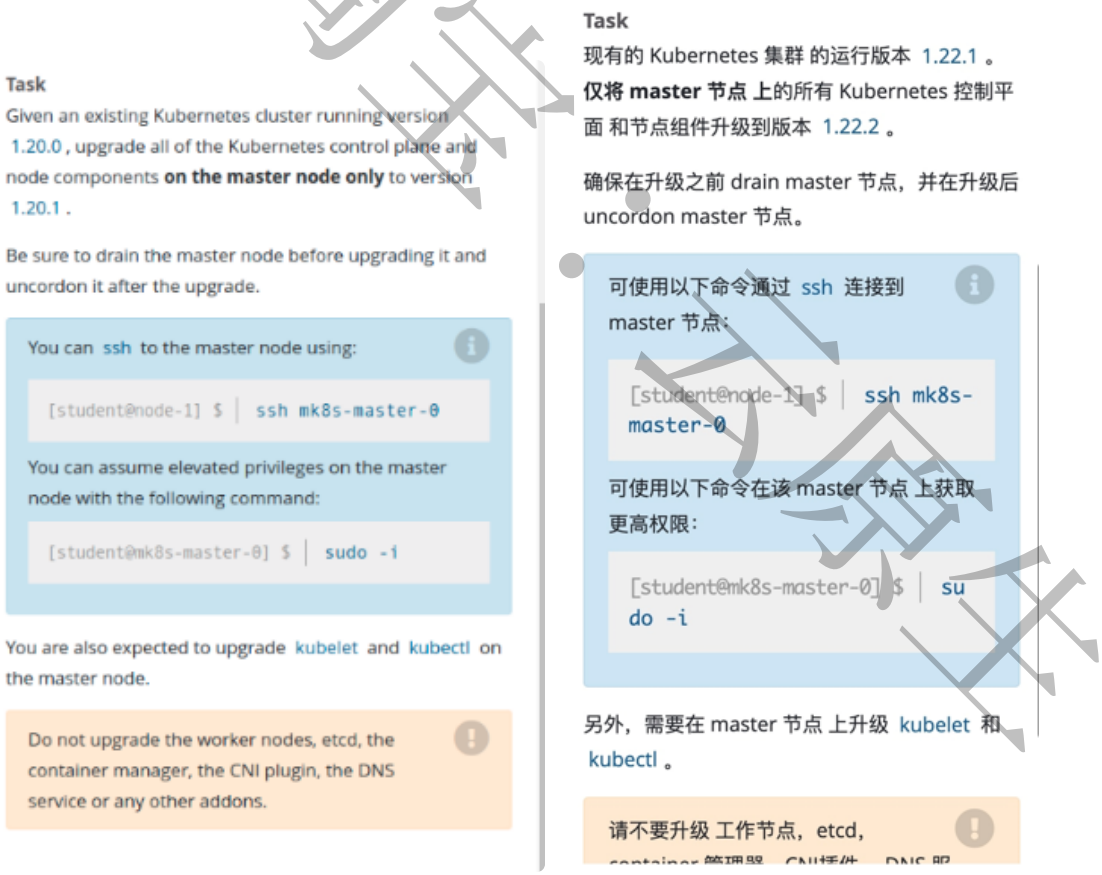
cordon节点，drain 节点，需要忽略 daemonsets 并清除 local-data，否则可能无法驱逐 pod

参考方法：

```
切换 context
kubectl cordon ek8s-node-1
kubectl drain ek8s-node-1 --ignore-daemonsets --delete-emptydir-data --force
完成后一定要通过 get nodes 加以确认
```

考点官网链接：https://kubernetes.io/zh/docs/tasks/administer-cluster/safely-drain-node/

第三题：升级 kubeadm



考题概述：

升级 master 节点为1.22.2，升级前确保drain master 节点，不要升级worker node 、容器 manager、 etcd、 CNI插件、DNS 等内容；

考题解析：

首先 cordon、drain master节点，其次升级 kubeadm 并 apply 到1.22.2版本，升级 kubelet 和 kubectl

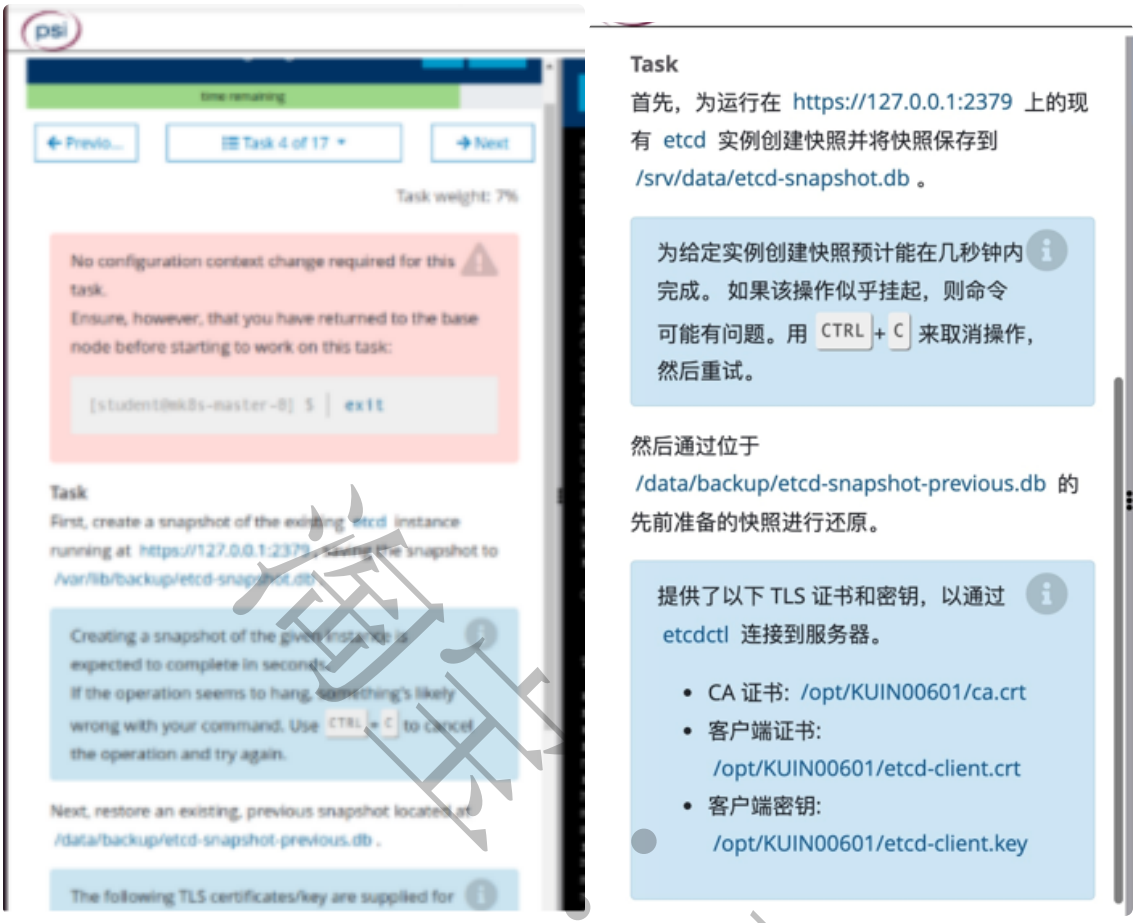
参考方法：

```shell
切换 context
kubectl get nodes
ssh mk8s-master-0
kubectl cordon mk8s-master-0
kubectl drain mk8s-master-0 --ignore-daemonsets --force
apt-mark unhold kubeadm kubectl kubelet
apt-get update && apt-get install -y kubeadm=1.22.2-00 kubelet=1.22.2-00 kubectl=1.22.2-00
apt-mark hold kubeadm kubectl kubelet
kubeadm upgrade plan
kubeadm upgrade apply v1.22.2 --etcd-upgrade=false
systemctl daemon-reload && systemctl restart kubelet
//kubectl -n kube-system rollout undo deployment coredns 有些朋友建议rollout coredns,
kubectl uncordon mk8s-master-0
```

注意：随着K8S版本的更新，升级的版本号也会有变化，只需调整升级命令对应的版本号即可；

考点官网链接：https://kubernetes.io/zh/docs/tasks/administer-cluster/kubeadm/kubeadm-upgrade/

第四题：备份还原 etcd



考题概述：

备份 https://127.0.0.1:2379 上的 etcd 数据到 /var/lib/backup/etcd-snapshot.db，使用之前的文件 /data/backup/etcd-snapshot-previous.db 还原 etcd，使用指定的 ca.crt 、 etcd-client.crt 、 etcd-client.key

考题解析：

备份 etcd 到指定目录、从指定备份文件还原 etcd

参考方法：

```shell
备份：
ETCDCTL_API=3 etcdctl --endpoints https://172.0.0.1:2379 --cacert=/opt/xxx/ca.crt --cert=/opt/xxx/etcd-client.crt --key=/opt/xxx/etcd-client.key snapshot save /var/lib/backup/etcd-snapshot.db
还原：
ETCDCTL_API=3 etcdctl --endpoints https://172.0.0.1:2379 --cacert=/opt/xxx/ca.crt --cert=/opt/xxx/etcd-client.crt --key=/opt/xxx/etcd-client.key snapshot restore /data/backup/etcd-snapshot-previous.db
还原成功后，最好通过 get nodes 确定集群状态是正常的

12月94分通过的朋友，etcd还原分享了一个思路，经测试可行，建议使用以下方式还原。（大概的测试方法：可以在etcd备份后单独创建一个pod容器，然后使用下面方法还原etcd后，get pod可以看到备份后创建的pod是没有恢复出来的。 ）

1.首先先将etcd、api停止了，移动静态pod文件后，过了一会容器会自动停止，
mv /etc/kubernetes/manifests /etc/kubernetes/manifests.bak
2.备份一下原来etcd的文件夹
mv /var/lib/etcd  /var/lib/etcd.bak
3.恢复数据
ETCDCTL_API=3 etcdctl --endpoints=https://127.0.0.1:2379 snapshot restore /data/backup/etcd-snapshot-previous.db --data-dir=/var/lib/etcd
4.启动etcd、api容器，把静态pod文件夹移回来 过一会就可以启动了
mv /etc/kubernetes/manifests.bak /etc/kubernetes/manifests
5.验证集群、pod资源状态
kubectl get nodes
kubectl get pods
```

考点官网链接：https://kubernetes.io/zh/docs/tasks/administer-cluster/configure-upgrade-etcd/

第五题：配置网络策略 NetworkPolicy

Task

在现有的 namespace fubar 中创建一个名为 allow-port-from-namespace 的新 NetworkPolicy 。

确保新的 NetworkPolicy 允许 namespace corp-net 中的 Pods 连接到 namespace fubar 中的 Pods 的端口 8080 。

进一步确保新的 NetworkPolicy :

- 不允许对没有在监听端口 8080 的 Pods 的访问
- 不允许非来自 namespace corp-net 中的 Pods 的访问

考题概述：

在命名空间 fubar 中创建网络策略 allow-port-from-namespace，只允许 ns my-app 中的 pod 连上 fubar 中 pod 的 80 端口，注意:这里有 2 个 ns ，一个为 fubar(目标pod的ns)，另外一个为 my-app(访问源pod的ns)

考题解析：

复制官网 services-networking/network-policies 中的案例，删掉不必要的部分，设置网络策略所属的 ns 为 fubar，端口为 80，设置 namespaceSelector 为源ns my-app 的labels

参考方法：

YAML 复制代码

```yaml
# kubectl get ns --show-labels

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-port-from-namespace
  namespace: fubar
spec:
  podSelector:
    matchLabels: {}
  policyTypes:
  - Ingress
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: my-app #  my-app命名空间标签
      podSelector:  # 此处podSelector前不要加 - ，加了则表示furbar 中的pod都可以访问furbar的80端口
        matchLabels: {}
    ports:
    - protocol: TCP
      port: 80
```

考点官网链接：https://kubernetes.io/zh/docs/concepts/services-networking/network-policies/#networkpolicy-resource

第六题：创建Service

考题概述：

重新配置已有的 deployment front-end，添加一个名称为 http 的端口，暴露80/TCP，创建名称为 front-end-svc 的 service，暴露容器的 http 端口，配置service 的类别为NodePort

考题解析：

按照需要edit deploy，添加端口信息，通过 expose 命令使用 NodePort 的方式暴露端口

参考方法：

YAML | 复制代码

```yaml
#题库练习创建deploy例子（考试系统deploy已存在）

apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: front-end
  name: front-end
spec:
  replicas: 1
  selector:
    matchLabels:
      app: front-end
  template:
    metadata:
      labels:
        app: front-end
    spec:
      containers:
      - image: nginx:1.21
        name: nginx
        ports:
        - containerPort: 80
          name: http
          protocol: TCP
```

YAML 复制代码

```
1) edit front-end , 在containers 中添加如下内容
kubectl edit deployment front-end
ports:
- name: http
  protocol: TCP
  containerPort: 80

2) expose 对应端口
kubectl expose deployment front-end --type=NodePort --port=80 --target-port=http --name=front-end-svc

验证:
# kubectl get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
front-end-svc   NodePort    10.105.156.36   <none>        80:32518/TCP   76s
root@k8s-master:~# curl  10.105.156.36

```

考点官网链接：https://kubernetes.io/docs/concepts/services-networking/service/

第七题：按要求创建 Ingress 资源

考题概述：

创建一个新的 Ingress 资源，名称 ping，命名空间 ing-internal，使用 /hello 路径暴露服务 hello 的 5678 端口

考题解析：

拷贝官文的 yaml 案例，修改相关参数即可，设置成功后需要通过 curl -kl <INTERNAL_IP>/hello 来测试

参考方法：

YAML 复制代码

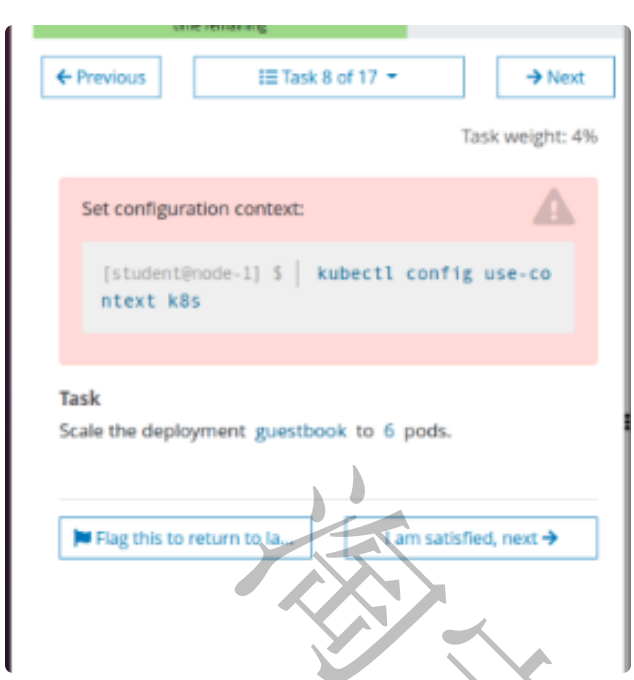
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ping
  namespace: ing-internal
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /hello
        pathType: Prefix
        backend:
          service:
            name: hello
            port:
              number: 5678
```

创建成功后， 通过get ingress 查看ingress的内外IP，然后通过提供的 curl 测试 ingress 是否正确，正常情况下是会输出 hello 的

```
root@k8s-master:~# kubectl get ingress -n ing-internal
NAME   CLASS    HOSTS   ADDRESS                                 PORTS   AGE
ping   <none>   *       192.168.123.151,192.168.123.152   80      109s
root@k8s-master:~# curl -kl 192.168.123.151/hello
hello
```

考点官网链接：https://kubernetes.io/zh/docs/concepts/services-networking/ingress/#the-ingress-resource

第八题：扩容Deployment



考题概述：

扩容 deployment guestbook 为 6个pod

考题解析：

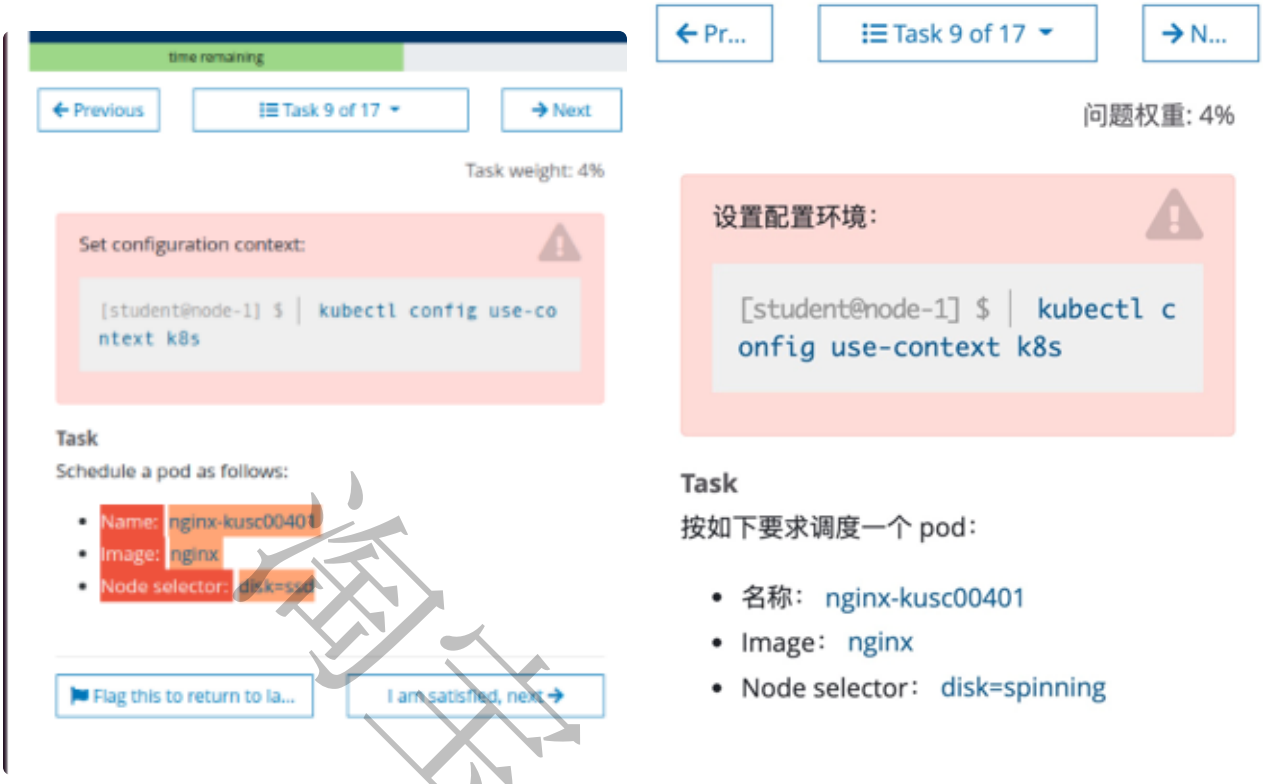
调整 replicas 为 6 即可，送分题

参考方法：

Shell | 复制代码

```
切换 context

kubectl scale deployment --replicas=6 guestbook
```

第九题：调度 pod 到指定节点



考题概述：

创建pod名称nginx-kusc0041，镜像nginx，调度该pod到disk=ssd的节点上

考题解析：

拷贝官文案例，修改下 pod 名称和镜像，删除多余的部分即可，送分题

参考方法：

YAML 复制代码

```
切换 context

apiVersion: v1
kind: Pod
metadata:
  name: nginx-kusc0041
spec:
  containers:
  - name: nginx
    image: nginx
  nodeSelector:
    disk: ssd
```

考点官网链接：https://kubernetes.io/zh/docs/concepts/scheduling-eviction/assign-pod-node/

第十题： 统计ready 状态节点数量

考题概述：

统计ready状态节点 要求不包括NoSchedule的节点

考题解析：

describe node过滤NoSchedule的节点，统计数量输入指定文档即可，送分题

参考方法：

```shell
kubectl describe nodes | grep -i Taints | grep -i -v NoSchedule | wc -l
echo number > /path/file
```

考点官网链接：https://kubernetes.io/zh/docs/concepts/scheduling-eviction/taint-and-toleration/

第十一题：创建多容器的pod

问题权重: 4%

考点概述:

创建名称为kucc1的pod，pod中运行nginx和redis两个容器

考点解析:

dry-run 一个pod，多追加一个镜像即可，送分题

参考方法:

Shell 复制代码

```
kubectl run kucc1 --image=nginx --dry-run=client -oyaml > 11.yaml
```

YAML 复制代码

```yaml
# 参考答案
apiVersion: v1
kind: Pod
metadata:
  name: kucc1
spec:
  containers:
  - name: nginx
    image: nginx
  - name: redis
    image: redis
```

考点官网链接：https://kubernetes.io/zh/docs/concepts/workloads/pods/

第十二题：按要求创建PV

考题概述：

创建一个名为app-config的PV，PV的容量为2Gi，访问模式为ReadWriteMany，volume的类型为hostPath，pv映射的hostPath为/srv/app-config目录

考题解析：

直接从官方拷贝合适的案例，修改参数，然后设置 hostPath为/srv/app-config即可

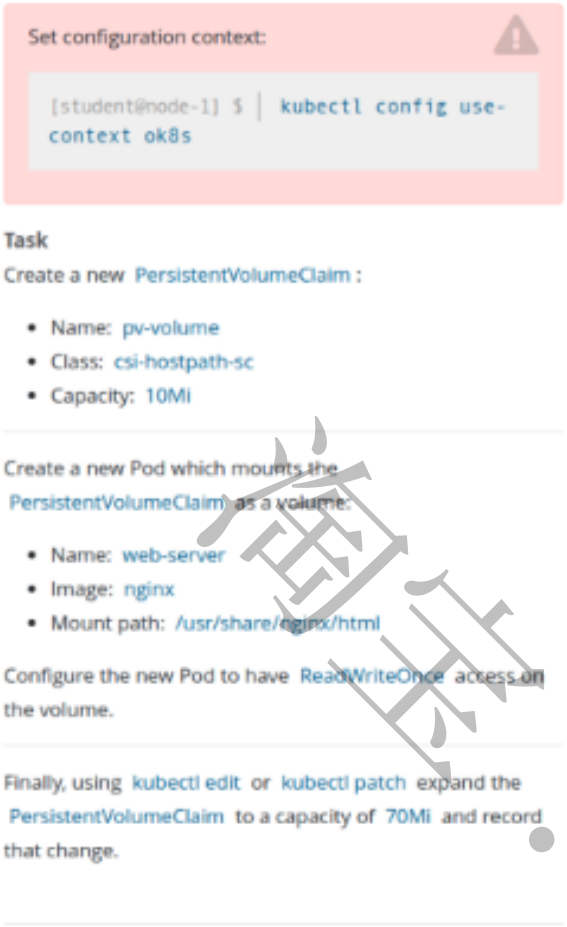
参考方法：

YAML 复制代码

```
切换 context

apiVersion: v1
kind: PersistentVolume
metadata:
  name: app-config
spec:
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: /srv/app-config
```

考点官网链接：https://kubernetes.io/zh/docs/tasks/configure-pod-container/configure-persistent-volume-storage/

第十三题：创建和使用PVC



Set configuration context:

```
[student@node-1] $ kubectl config use-context ok8s
```

Task

Create a new PersistentVolumeClaim :

- Name: pv-volume
- Class: csi-hostpath-sc
- Capacity: 10Mi

Create a new Pod which mounts the PersistentVolumeClaim as a volume:

- Name: web-server
- Image: nginx
- Mount path: /usr/share/nginx/html

Configure the new Pod to have ReadWriteOnce access on the volume.

Finally, using kubectl edit or kubectl patch expand the PersistentVolumeClaim to a capacity of 70Mi and record that change.

Task

创建一个新的 PersistentVolumeClaim：

- 名称: pv-volume
- Class: csi-hostpath-sc
- 容量: 10Mi

创建一个新的 Pod，来将 PersistentVolumeClaim 作为 volume 进行挂载：

- 名称： web-server
- Image： nginx
- 挂载路径： /usr/share/nginx/html

配置此新 Pod，使得对上述 volume 具有 ReadWriteOnce 权限。

最后，使用 kubectl edit 或 kubectl patch 将 PersistentVolumeClaim 的容量扩展为 70Mi，并记录此更改。

考题概述：

使用指定storageclass csi-hostpath-sc创建一个名称为pv-volume的 pvc，容量为10Mi 创建名称为web-server的pod，将nginx 容器的/usr/share/nginx/html目录使用该pvc挂载 将上述pvc的大小从10Mi更新为70Mi，并记录本次变更；

考题解析：

根据官方文档拷贝一个PVC，修改参数，复制官网nginx 的pod，然后添加volumeMounts和volume

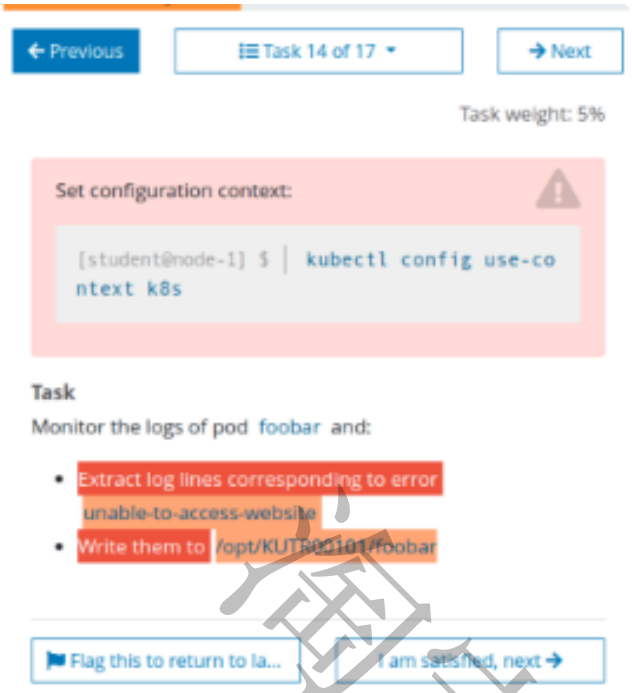
参考方法：

YAML 复制代码

```
切换 context

# vim 13.1.yaml

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pv-volume
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Mi
  storageClassName: csi-hostpath-sc

vim 13.2.yaml
apiVersion: v1
kind: Pod
metadata:
  name: web-server
spec:
  containers:
    - name: nginx
      image: nginx
      volumeMounts:
      - mountPath: "/usr/share/nginx/html"
        name: mypv
  volumes:
    - name: mypv
      persistentVolumeClaim:
        claimName: pv-volume

# kubectl apply -f 13.yaml
# kubectl edit pvc pv-volume --record
更改 10Mi 为 70Mi

```


考点官网链接：https://kubernetes.io/zh/docs/tasks/configure-pod-container/configure-persistent-volume-storage/

第十四题：监控pod的日志



考题概述：

监控foobar pod中的日志

获取包含unable-to-access-website的日志，并将日志写入到/opt/KUTR00101/foobar

考题解析：

kubectl logs 获取日志，通过 grep 进一步获取目标日志，送分题

参考方法：

YAML 复制代码

```
切换 context

kubectl logs foobar | grep unable-to-access-website > /opt/KUTR00101/foobar
```

第十五题：添加 sidecar 容器并输出日志

考题概述：

添加一个sidecar容器(使用busybox 镜像)到已有的pod 11-factor-app中，确保sidecar容器能够输出/var/log/11-factor-app.log的信息，使用volume挂载/var/log目录，确保sidecar能访问11-factor-app.log 文件

考题解析：

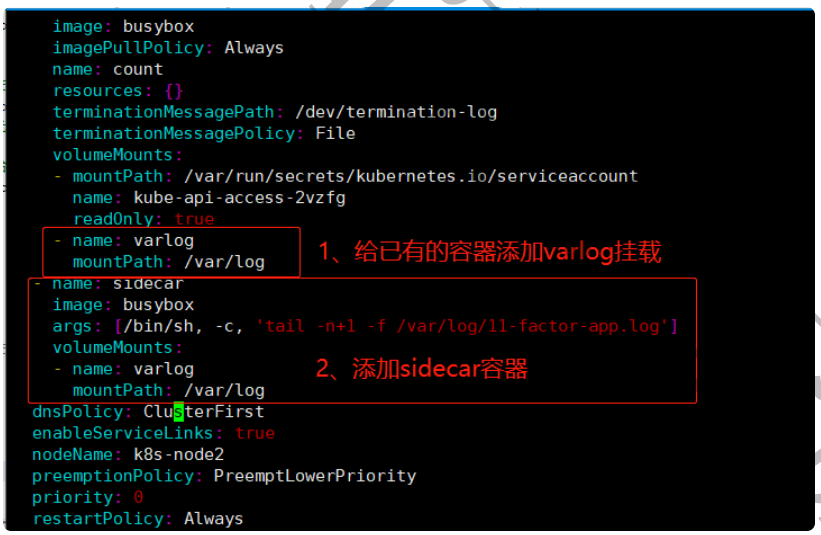
通过 kubectl get pod -o yaml 的方法备份原始 pod 信息，删除旧的pod 11-factor-app copy 一份新 yaml 文件，添加 一个名称为 sidecar 的容器，新建 emptyDir 的卷，确保两个容器都挂载了 /var/log 目录，新建含有 sidecar 的 pod，并通过 kubectl logs 验证；

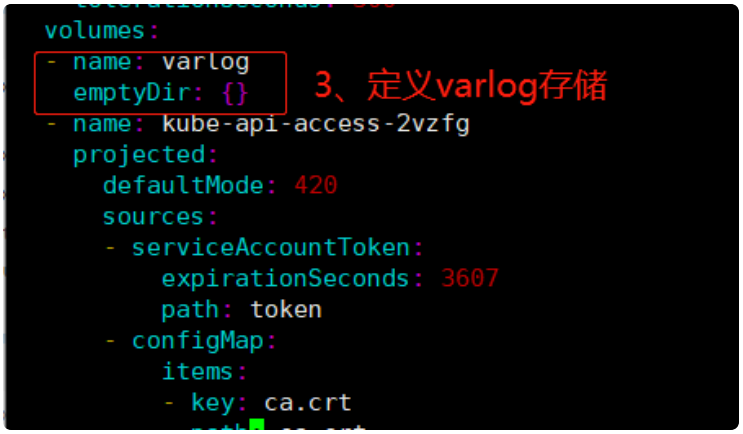
参考方法：

YAML 复制代码

```
答题思路：
  先用kubectl get podname -o yaml > podname.yaml 获取到yaml文件，然后删除旧的pod；
再重新 copy 一个新的 yaml 添加 sidecar 容器，并在两个容器中都挂载 emtpyDir 到 /var/log目录，最后通过apply 生成带 sidecar 的 pod；pod 正常拉起后,通过 kubectl logs 11-factor-app sidecar 确认能正常输入日志即可

备份原始pod信息
# kubectl get pod 11-factor-app -oyaml > sidecar-new.yaml
# cp sidecar-new.yaml 15.yaml
删除pod
# kubectl delete pod 11-factor-app 或者 kubectl delete -f sidecar-new.yaml
```

编辑yaml 添加sidecar容器以及给已有的容器添加挂载：





```shell
kubectl apply -f 15.yaml
# 要保证pod中两个容器都是Running状态
root@k8s-master:~# kubectl get po 11-factor-app
NAME            READY   STATUS    RESTARTS   AGE
11-factor-app   2/2     Running   0          112s
kubectl logs 11-factor-app sidecar  #验证日志输出
0: Thu Dec 23 15:15:50 UTC 2021
0: Thu Dec 23 15:15:55 UTC 2021
0: Thu Dec 23 15:16:00 UTC 2021
0: Thu Dec 23 15:16:05 UTC 2021
0: Thu Dec 23 15:16:10 UTC 2021
0: Thu Dec 23 15:16:15 UTC 2021
0: Thu Dec 23 15:16:20 UTC 2021
0: Thu Dec 23 15:16:25 UTC 2021
0: Thu Dec 23 15:16:30 UTC 2021
0: Thu Dec 23 15:16:35 UTC 2021
0: Thu Dec 23 15:16:40 UTC 2021
0: Thu Dec 23 15:16:45 UTC 2021
0: Thu Dec 23 15:16:50 UTC 2021
0: Thu Dec 23 15:16:55 UTC 2021
```

注意：拷贝官网案例，这题要注意复制粘贴的位置，因为有几个朋友反馈添加Sidecar容器后pod启动不了、或者格式有问题。

考题官网书签：https://kubernetes.io/zh/docs/concepts/cluster-administration/logging/

第十六题：查看 cpu 使用率最高的 pod

考题概述：

查找label为name=cpu-loader的pod，筛选出cpu负载最高的那个pod，并将名称追加到/opt/KUTR00401/KUTR00401.txt

考题解析：

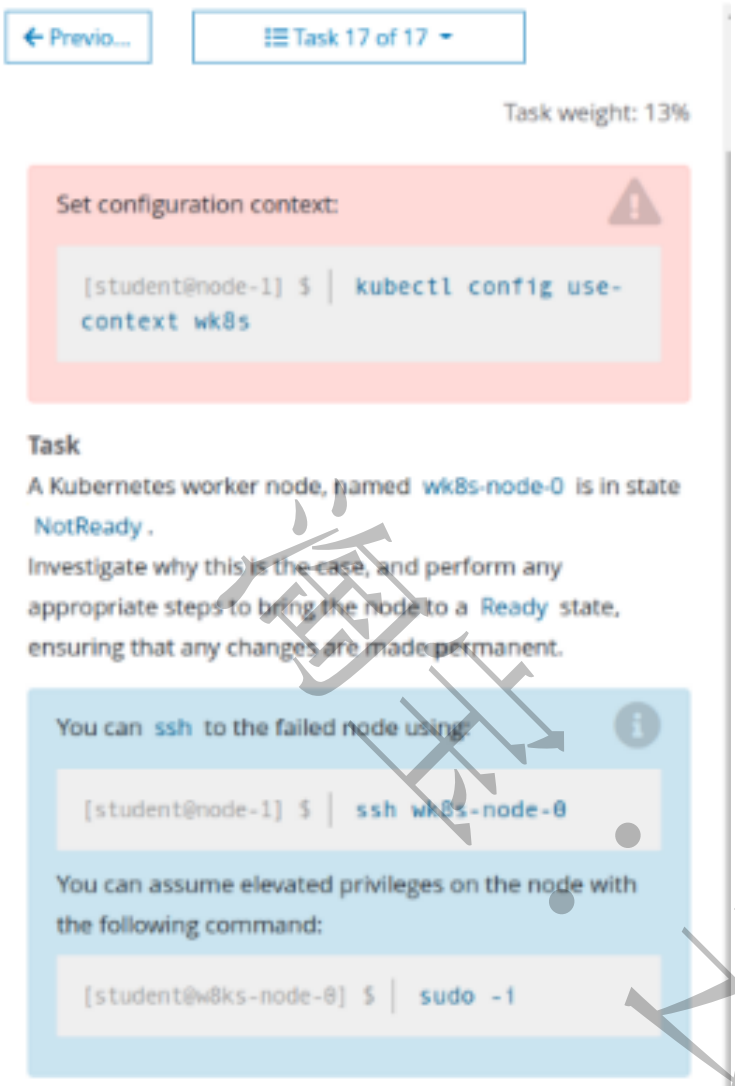
使用top命令，结合 -l label_key=label_value 和 --sort=cpu 过滤出目标即可

参考方法：

Shell 复制代码

```
切换 context

kubectl top pod -l name=cpu-loader -A --sort-by='cpu'
echo podName >> /opt/KUTR00401/KUTR00401.txt
```

第十七题：排查集群中故障节点



考题概述：

节点wk8s–node–0状态为NotReady，查看原因并恢复其状态为Ready

确保操作为持久的

考题分析：

通过get nodes查看异常节点，登录节点查看kubelet等组件的status并判断原因

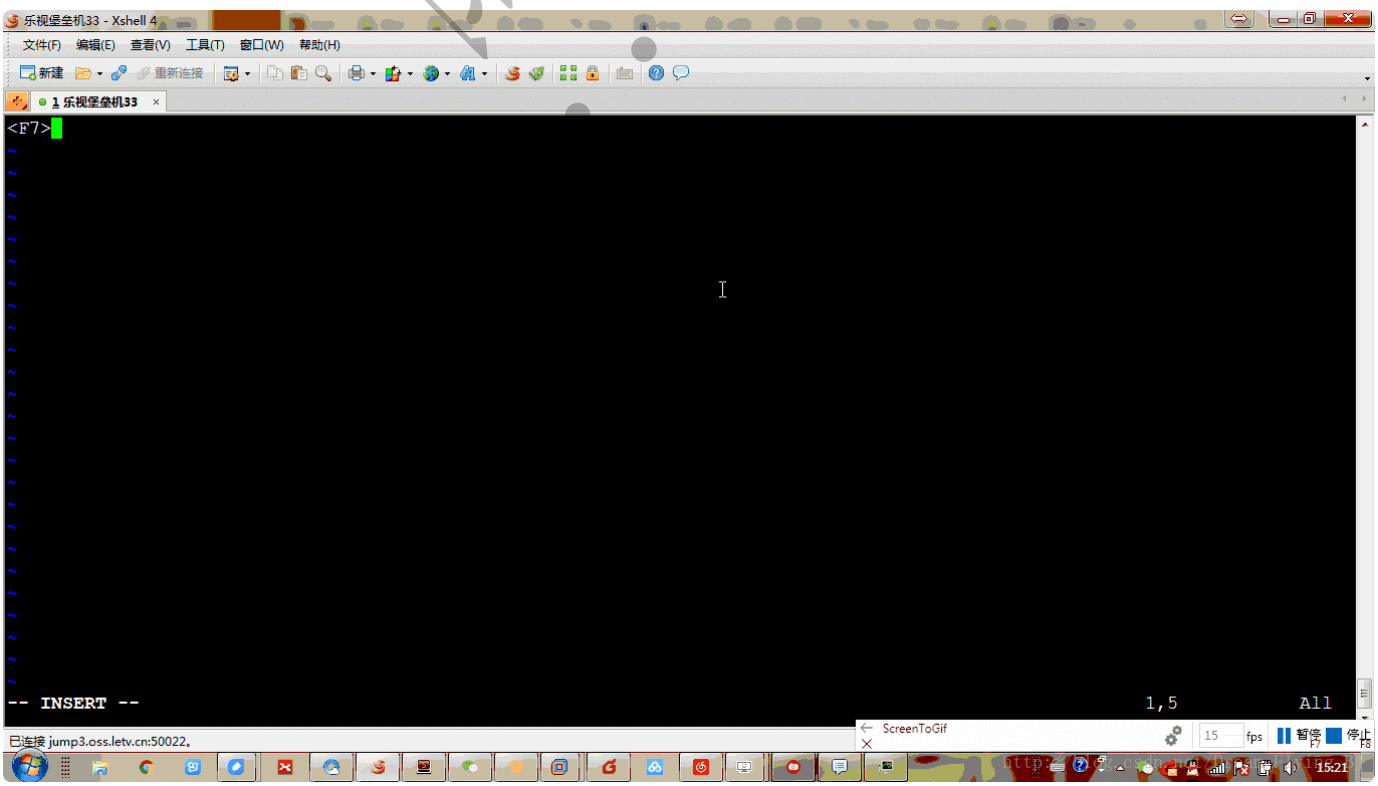
启动kubelet并enable kubelet即可

参考方法：

```shell
切换 context

kubectl get nodes
ssh wk8s-node-0
sudo -i
systemctl status kubelet
systemctl enable kubelet
systemctl restart kubelet
systemctl status kubelet

再次 get nodes,  确保节点恢复 Ready 状态
```


附录一、paste模式设置

先设置paste模式后，再进行粘贴，这样就不用再手动调整格式、缩进问题了。



如果PDF文件不显示动图效果，可以通过这个地址查看：https://www.yuque.com/docs/share/e0949e2f-b053-4bbf-8a02-d161f3f3b32a?# 《paste模式设置（动图）》

感兴趣的朋友也可以加Q群，后续题库更新会第一时间通知，并不定时分享各种云原生技术资源，一起学习、一起交流、一起进步；