- CKAD Specific
  - setup
    - alias k='kubectl'
  - mapping
    - ConfigMap
      - as env
        - spec.containers[0].env[0].valueFrom.configMapKeyRef.name
        - spec.containers[0].envFrom.configMapRef.name
      - as volume
        - spec.containers.volumeMounts.mountPath
        - spec.volumes.configMap.name
    - Secret
      - as volume
        - spec.containers.volumeMounts.mountPath
        - spec.volumes.secret.secretName
    - SecurityContext
      - spec.securityContext.runAsUser
      - spec.securityContext.capabilities.add: ['<capability>']
    - Volumes
      - shared empty dir
        - spec.containers.volumeMounts.mountPath
        - spec.volumes.emptyDir: {}
    - PersistentVolume
      - spec.containers.volumeMounts.mountPath
      - spec.volumes.persistentVolumeClaim.claimName: <name>
    - ServiceAccount
      - spec.serviceAccountName: <name>
  - debugging
    - wget -O- <ip>:<port>
    - (decodes a secret) echo YWRtaW4K | base64 -d
    - (check a mount) mount | grep <name>
    - k get pod <name> | grep -i annotations
    - k get po <name> -o jsonpath='{.spec.containers[].image}{"\n"}'
  - kubectl commands
    - create
      - ConfigMap
        - k create cm <name>
          - --from-literal=<key>=<value>
          - --from-env-file=<file>
          - --from-file=<name>=<file>
      - Secrets
        - k create secret generic <name>
          - --from-literal=<key>=<value>
          - --from-file=<filename>
      - Service
        - k create svc nodeport <name> --tcp=<servicePort>:<targetPort>
          - --dry-run -o=yaml
          - (defaults selector to app: <name>
      - NetworkPolicy
        - k create -f policy.yaml (only creates from yaml
      - ServiceAccount
        - k create sa '<user>'
      - PersistentVolume
        - k create -f pv.yaml (only creates from yaml
      - Deployment
        - k create deploy <name> --image=<image> defaults selector to app: <name>
    - run
      - Pod
        - k run <name> --restart=Never --image=<image>
          - -- <command> /bin/sh -c 'sleep 3600' -c read command from following stri
          - --dry-run -o=yaml
          - -n <namespace>
          - --command '<command>'
            - (note no =)
            - command: ['/bin/sh', '-c', 'echo ...']
          - --env=<name>=<value>
          - --port=80
          - --expose
            - (create service as well)
            - (clusterIP default)
          - --serviceaccount=<user> (check for <user>-token in secrets
          - --requests='cpu=100m,memory=256Mi' --limits='cpu=200m,memory=512Mi'
        - k run <name> --rm --image=<image> -it --restart=Never -- sh
          - (rm only works in it container
          - example yaml commands
            - command: ["nginx"]
            - args: ["-g", "daemon off;"]
      - Job
        - k run <name> --restart=OnFailure --image=<image>
        - spec.completions=5 (add number of times)
        - spec.parallelism=5 (run in parallel)
      - CronJob
        - k run <name> --restart=OnFailure --image=<image> --schedule="*/1 * * * *"
      - Deployment
        - k run <name> --restart=Always --image=<image>
          - --replicas=<replicas>
          - --port=<port> (creates container port)
          - --expose (creates a service as well)
          - --labels=<key>=<value>
          - --dry-run -o=yaml
          - (defaults selector to run: <name> if run command use
    - annotate
      - k annotate po <name> <key>=<value> --overwrite
      - k annotate pp <name> <key>- (remove)
    - autoscale
      - k autoscale deploy <name> --min=<minNum> --max=<maxNum> --cpu-percent=<percent>
    - copy
      - k cp <name>:<mountPath> .
    - delete
      - k delete <resource> <name>
    - describe
      - k describe <resource> <name>
    - edit
      - k edit <resource> <name>
    - exec
      - k exec -it <name> -- <command>
    - expose
      - k expose deploy foo --port=<servicePort> --target-port=<targetPort>
      - explain
        - k explain <resource>.spec
    - get
      - k get po
        - --all-namespaces
        - --show-labels
        - --o=wide (show more details inclusing i
      - k get po -l <key>=<value> (filtered by label)
      - k get po -L <key> (list with label)
      - k get po --selector=<key>=<value>
      - k get ep (endpoints)
      - k get po -o=wide (show ports etc.)
      - k get rs
      - k get <resource> <name> -o=yaml --export (export without start info
      - k get jobs -w (wait)
    - label
      - k label po <name> <key>=<value> --overwrite
      - k label po <name> <key>-
      - k label po <name1> <name2> <name3> <key>- (remove from multiple
      - k label po <name>{1..3} <key>- (remove from multiple
    - logs
      - k logs <name>
        - -f (follow)
        - -c=<container> (container)
        - -p (previous instance)
    - rollout
      - k rollout status deploy <name>
      - k rollout history deploy <name> --revision=<revision>
      - k rollout undo deploy <name> --to-revision=<revision>
      - k rollout pause deploy <name>
      - k rollout resume deploy <name>
    - scale
      - k scale deploy <name> --replicas=<num>
    - set
      - k set image deploy <name> <container>=<image>
    - **Use k <command> <option> --help to see all options**