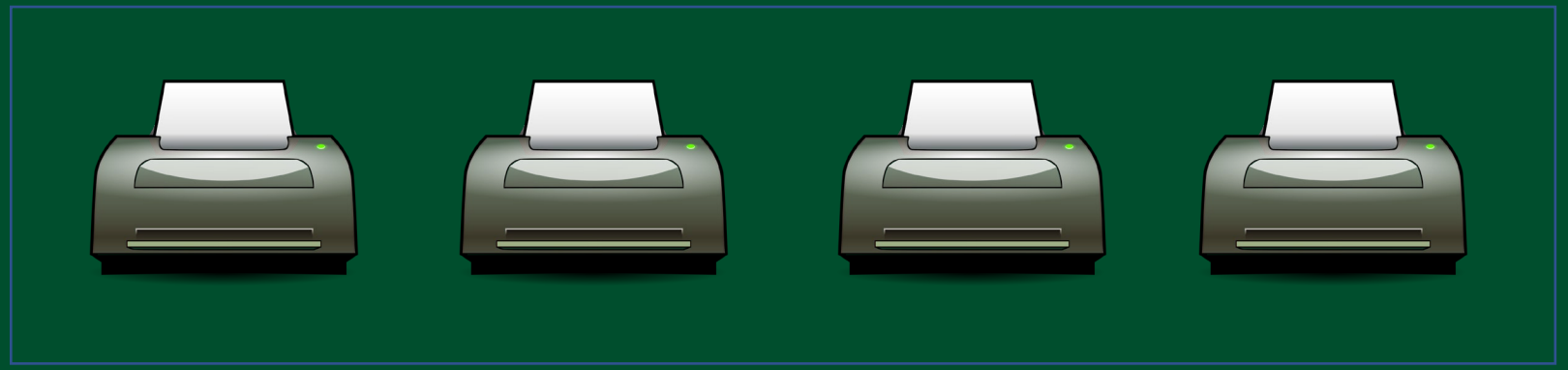
Exercise 1: Interfaces

- Consider a service for concurrently sharing a set of printers from a pool for several users (all printers are equal). Define an appropriate interface (for instance using Python-like oder C-like pseudocode) for such a service so that the openness requirements are satistified.
- Assume that the channel is secure.



```
def Print(printerID:int, data: Document, settings:Settings, tokem: int) ->int:
# 0 = OK
# -1 = not authorised
# -2 = queue full
# -3 = not available
# -4 = printer doesn't exist
# -5 = just error

def AuthentificateUser(userID:int, password:str) -> int:
# token if correct
# -1 if not correct

def CreateUser(username:str, password:str) -> int:
# return userID

def LogOut(token:int, userID: int) -> int:

def GetQueueLenght(printerID, int, token: int) -> int:#queue len

def GetPrinterID(token:int) ->list[int]

def addPrinter(printerInfo:PrinterInfo, token:int) -> int:

def getPrinterInfo(token:int, printerID:int) - > PrinterInfo

def removePrinter(printerID:int, token:int, reason:str) ->int
#0 = ok
#1=no printer
#2 = general error
# 3 = not authorised
```

Exercise 2: Transparency

- Consider a system with the following requirements: a supercomputing cluster offers high-performance computing services to a scientific organization. This organization (and the cluster, too) is geographically distributed across different countries in several continents.
- The system must be capable of running batch computing jobs sent by its clients and reporting the results (output files).
- List what types of transparencies are desirable and which not. Explain the reasons behind your choices.

Access	Location	Migration	Replication	Relocation	Concurrency	Fault

Exercise 3: Pervasive systems

- Take a look at your environment. Make a list of all pervasive computing systems you have interacted with in the last month.
- For each system, try to infer what devices are involved and which data might be used as input to the system.