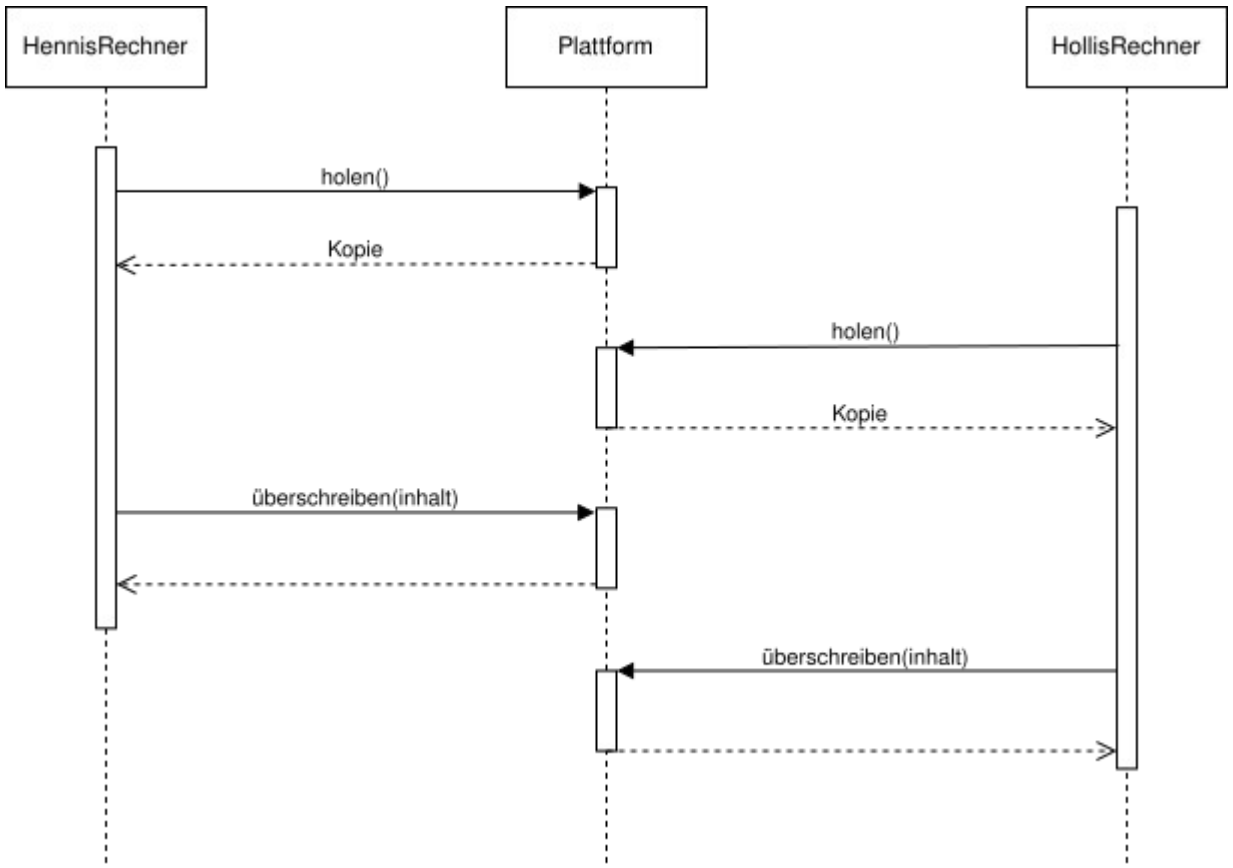
Q12; Schulaufgabe Informatik; 23.02.24; Beispiellösung

1.

a)



(3 BE)

Henni und Holli laden nacheinander dieselbe Version der Datei herunter und bearbeiten sie unabhängig voneinander. Beim Hochladen überschreibt Holli allerdings die von Henni überarbeitete Version der Datei mit seiner überarbeiteten Version der Datei (in der Hennis Änderungen nicht enthalten sind) => Hennis Änderungen gehen also verloren – nur Hollis Änderungen bleiben erhalten.
(2 BE)

b)

Ein solcher „Mechanismus“ müsste einzelnen Nutzenden die Möglichkeit bieten, den Zugriff auf eine Datei für andere Nutzende zu sperren (und später wieder freizugeben). Bevor sie selbst zugreifen, müssten Nutzende diese Sperre für die betreffende Datei aktivieren (sofern möglich – ansonsten müssten sie warten, bis das möglich ist). Nachdem die Bearbeitung der Datei abgeschlossen ist und die veränderte Datei hochgeladen ist, müsste die Sperre wieder aufgehoben werden.
(2 BE)

2.

a)

Erzeuger-Verbraucherproblem:
Ein Roboter wartet auf neue Aufträge, die aber nicht erteilt werden können, weil dazu noch die Daten von weiteren Robotern nötig wären – die aber nicht andocken können, weil sie vom ersten blockiert werden. (3 BE)

b) Im Diagramm wird in die Warteschleife eingefügt: „von der Basisstation abdocken und beiseitefahren“, („festgelegte Zeit warten“), „zur Basisstation fahren und andocken, sobald diese frei ist“ (4 BE)

c) Vergleich: Abdocken entspricht momentanem Freigeben einer Ressource (wait)
Roboter dockt aber selbstständig wieder an, um nach neuen Aufträgen zu sehen, während ein wartender Thread erst wieder aktiv wird, wenn er entsprechend benachrichtigt wird (notify)
(2 BE)

3.
a)

	AK	PC	100	101	102
	/	0	5	5	/
LOAD 100	5	2	5	5	/
CMP 101	5	4	5	5	/
JMPZ 12	5	12	5	5	/
LOADI 1	1	14	5	5	/
STORE 102	1	16	5	5	1
HOLD	1	18	5	5	1

(4 BE – 1 BE Abzug pro fehlerhafte Tabellenzeile <=> nicht weniger als 0 BE)

b)
```
fun aufGleichheitPruefen(wert1: Int, wert2: Int): Int { // 1 BE
  if (wert1 == wert 2) return 1 // 1,5 BE
  else return 0 // 0,5 BE
}
```

c)
```
LOAD 100
STORE 102
LOAD 101
SUBI 1
JMPZ 20
STORE 101
LOAD 102
MUL 100
STORE 102
JMP 4
HOLD
```

Startwert für 102: 1 BE
101 laden und schreiben: 1 BE
1 subtrahieren: 1 BE
ggf. springen: 1 BE
102 laden und schreiben: 1 BE
multiplizieren: 1 BE
zurückspringen: 1 BE
HOLD: 1 BE

4.
a)
Im Diagramm werden einander zugeordnet: Control – Steuerwerk, Processing unit (ALU) – Rechenwerk, Input interface port – Eingabewerk, Output interface port – Ausgabewerk. Werden die beiden Letzteren zu Ein-Ausgabewerk zusammengefasst, muss Data highway und Program highway der Begriff Bussystem zugeordnet werden. (4 BE)
Unterschied: Von-Neumann: ein Speicherwerk / ein Bussystem ↔ Harvard: getrennte Speicher für Programm und Daten mit zugehörigen getrennten Bussystemen (2 BE)

b)
Schritt „Befehl holen“ kann gleichzeitig mit „Befehl ausführen“ stattfinden, da aufgrund der getrennten Bussysteme Ersterer durch Zugriff auf den Programmspeicher, Letzterer parallel dazu durch Zugriff auf den Datenspeicher realisiert werden kann ↔ in der Von-Neumann-Architektur gibt es aber nur ein Bussystem, über das nur entweder Programmcode (von einer Speicheradresse) oder ein Datenwert (von einer anderen Speicheradresse) geladen werden kann. (3 BE)