

QAD 5
④ public

int dequeue() {

if (S2.isEmpty())

$O(1)$

while (!S1.isEmpty())

$O(n)$

S2.push(S1.pop());

$O(1)$

return S2.pop();

$O(1)$

} $O(n)$

GAD5 1

(5) (a) ~~if (s2.is~~

(b) Ich definiere, dass für jedes enqueue und dequeue man +4 Punkte bekommt und für jede Operation verbraucht man einen Punkt. Also nach einem enqueue bekommt man immer +3 eingezahl auf das Konto und bei dequeue gibt's zwei Fälle:

1. $s2.isEmpty() = false$

Dann bekommt man +2 am Ende.

2. $!s2.isEmpty()$

Dann wird die Schleife n -mal ausgeführt:
 $-3n - 2 + 4 = \underline{-3n + 2}$.

Betrachten wir den worst case: es wurde n -mal enqueued und dann will man dequeue machen:
 $3n - 3n + 2 = 2 > 0$

Also es das Bankkonto ist immer Positiv.

$$A(\text{enqueue}) = \cancel{O(1)} + O(3) = O(1)$$

$$A(\text{dequeue}) = 3n + 2 - 3n + 2 = 4 = O(1)$$

Nun hat man immer ein const Laufzeit.

6

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(3)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(42)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(51)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(17)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(29)
29	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(23)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(24)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	D(9)
6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(18)
6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(40)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(14)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	D(23)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	I(23)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	D(40)
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	