Tema 2 TPM
Manolache Claudiu Grupa TPM4, 3E3
1.
a) pred.next = node;
b) if (key == current.key){ return false;}
c)if (key == current.key) { // present
  pred.next = current.next;
d)if (key == current.key) { // present
  pred.next = current.next;
  return true;
 } else {
  return false; // not present
 }

linia cu return false:

2.
#atomicInteger = integer care garanteaza increase si decrease in timp util
2 a)Grupele TPM3 si TPM4: Mai este necesara in metoda *deq* plasarea *size.getAndDecrement()* in cadrul sectiunii protejate de *deqLock*? Argumentati.
Nu este necesara, pentru ca size.getAndDecrement() se intampla intr-o cuanta de timp pentru ca size este de tip atomicInteger si asigura ca valoarea la atomic integer e corecta (nu e modificata in acelasi timp de procese/thread-uri diferite).
b)Grupele TPM3 si TPM4: Va mai functiona in acest caz metoda *deq* corect pastrand caracterul FIFO al cozii? Argumentati.

Da, pentru ca modul in care se fac lock-urile pentru head si tail functioneaza la fel ca in codul de la a pentru enq si deq.
c)
d)

Da, este necesara, pentru ca 2 thread uri verifica simultan daca coada este empty sau nu si daca coada are un singur element, atunci un thread scoate elementul din coada si celelalt o sa dea eroare, iar prezenta lock ul la deq asigura ca un singur thread face operatiile intr-un interval de timp

3.

	100 000 elemente		
	time(ms)	startTime	stopTime
add	1	1641501917855	1641501917856
remove	1	1641501982760	1641501982761
contains	1	1641502095268	1641502095269