8. find_cheap_tic(p, t_event, t_row) link list

```
cheapest = NULL
for each ticket in db list:                    n times
    if ticket has correct event and row less than t_row:
        else:                                  O(1)
            if cheapest.price > ticket.price:
                cheapest = ticket
return cheapest.
```

⇒ O(N). b/c have to loop thru entire dB (once)

9. remove_expensive_tic (p, event)

```
row_num = 0
for each ticket in db:                         O(N)
    if ticket.event = event and ticket.row > row_num;
        row.num = ticket.row
for row starting at 1 to row_num               O(N)
        price = 0
        tic = 0.
    for each ticket in db                      O(N)
        if ticket.event = event and ticket.row = row
            price += ticket.price
            tic++
    average = price / tic
```

O(N) [for each ticket in db
    if right ticket price = 2 · average price.
        delete.

$O(N) + O(M(N+N))$

$= O(N) + O(MN)$

$= O(M)$ b/c $M << N$.

- function loops all entry db, once all beginning loops twice per row but row << db size.

3. No. line 23 is simply incrementing a pointer, not accessing anything (other than the memory address for p itself), so it doesn't matter if the memory location is valid or not.

1- missing <return value> in int main()

missing <junk> in other spaces.

MC. 5. C — miss <return value>.

6. d.

8. C