Problem 4):-

5 delivery guys, further address prioritized.

a)-

```
condition avail_car;
Semaphore mutex = 1;

void request_delivery(int order_id, int distance, int size)
{
    waiting_orders[num_waiting].id = order_id;
    waiting_orders[num_waiting].dist = distance;

    wait(mutex);
    num_waiting++;
    signal(mutex);
    waiting_orders.sort();          // sorts in descending order
    while(waiting_orders[0].dist != distance ||   // or
          (waiting_orders[0].dist == distance && available_cars == 0))
        wait(avail_car)

    available_cars--;
    wait(mutex);
    num_waiting--;
    signal(mutex);
}

void release_car(){
    available_cars++;
    broadcast(avail_car);
}
```

b)- Prioritization of distance policy creates risk of bankruptcy. Because some short distance order might never be delivered and number of them increases by time. Let's say all cars are busy and we got 3 new orders (2 long, 1 short). Every available car will deliver one of the long distance from waiting queue. Short distance might never be delivered company delivers 2 of the every 3 orders. So, yes there is risk of bankruptcy.

c)-

```
void request_delivery(int order_id, int distance, int size) {
    waiting_orders[num_waiting].id = order_id;
    waiting_orders[num_waiting].dist = distance;
    waiting_orders[num_waiting].size = size;
    wait(mutex);
    num_waiting ++;
    signal(mutex);
    waiting_orders.sort();
    while(waiting_orders[0].dist != distance ||
          (waiting_orders[0].dist == distance && available_cars < size)
        wait(avail_car)
    available_cars -= size;
    wait(mutex)
    num_waiting --;
    signal(mutex);
}
```

d)- Yes, it increases the risk of bankruptcy. Because when we receive long distance order with size big enough, then all cars are busy with only 1 delivery and all other orders may not be delivered on time causes the lose money.