**Assumptions:**

1. Every event would surely have a ticket.
2. Every ticket would surely have a quantity and price associated with it.
3. Every Event has a location and there can be only one location for an event.
4. The minimum value of the ticket would be returned even if the quantity of the ticket is 0.
5. The program generates 100 events every time the program is compiled and run.

**1.How might you change your program if you needed to support multiple events at the  same location?**

My design supports for an event to have multiple locations. Thus I would not have changed anything in my program.

**2.How would you change your program if you were working with a much larger world size?**

* To find the nearest 5 events, I would divide the map into smaller sectors (the length would be 10 units in both the dimensions). Every sector would be assigned to a list of events. Based on the sector in which the user is in, the distance of the events in those sectors are calculated and returned. If the user is at the edge one sector, then the neighbouring sector is also taken for the calculation.
* I would index every range of the nearest location inside the sector. For example, if Event A takes place at (0,0) and Event B takes place at (0,10), the location 0 through 5 would be index for location A and the location 5 through 10 would be index for location B. Hence if the user is in the location 3, it would immediately return the Event A.
* A heap would be used to store the ticket prices for ever event. Hence when the minimum ticket price is requested, the top of the heap would be peeked and returned. This would improve performance drastically.
* I would use UUID instead of int as a data type for the events.