1. ***Select a non-trivial method in one class that can be invoked by another class.  [Creation/deletion methods are usually trivial.]***

Run(), which is found in the Task class, but can be invoked by the User class.

1. ***Write up a method specification for it, along the lines of the example at the end of the updated Ch.8 slides (discussed in class on 11/12/18).***

**Method Name:** Run()

**Class Name:** Task

**ID:** TA-RU-R

**Clients:** User

**Associated Use Cases:** TasksRunning, Failed, Success

**Description of Responsibilities:** Create a session on TicketMaster.com and search for tickets based on the given keyword. When the ticket is found, immediately checkout using the given billing information.

**Arguments Received:** Keyword, BillingInfo

**Type of Value Returned:** Boolean, along with a string that describes what happened

**Pre-conditions:** BillingInfo must contain a valid credit card, which would be checked using a verification system that ensures there are a correct amount of numbers. There is no pre-condition on the keyword, as it can be anything the user wants.

**Post-conditions:** If the ticket was successfully purchased, then run success() which tells the user that their card will be charged due to a successful purchase. If the ticket was not purchased after 5 minutes, then run failed(), which should display why the ticket was not purchased (sold out, keyword not found, etc.)

**Algorithm:**

* Use the Ticketmaster Mobile API to view a list of all tickets
* Automatically refresh every 200ms until the tickets with the keyword are loaded in stock
* Select the ticket, and select the best general seat available
* At checkout, send a HTTP request with the user’s billingInfo as the argument in order to immediately checkout
* Click checkout and wait for TicketMaster to return a page
* If the ticket confirmation is returned, run success() and let the user know. If another page is returned, use that page to display why the purchase failed when running failed()

**Side-effect:** Return whether or not the ticket was purchased along with a reason if it wasn’t/

***3.Sketch an alternative algorithm for the same example.***

**- Open a browser on the user’s computer and go to TicketMaster.com**

**- Go to the View All section and enter the user’s keyword in the search bar**

**- Automatically keep clicking search every 800ms until the ticket is found**

**-** Select the ticket, and select the best general seat available

-At checkout, autofill the information into the given fields based on the user’s billing information.

- Click checkout and wait for TicketMaster to return a page

- If the ticket confirmation is returned, run success() and let the user know. If another page is returned, use that page to display why the purchase failed when running failed()

***4. Say why either algorithm is better or preferable, identifying possible trade-offs.***

**The first algorithm is much faster, as it can find tickets quickly due to the low refresh time and can checkout by simply using an HTTP request. However, it is more prone to TicketMaster’s bot detection, which can cancel orders based on whether or not they determine that the user was assisted by a third-party. This is the benefit of the second algorithm; even though it is considerably slower, it is still faster than what a manual user can do, but it is almost impossible to be caught by bot detection, as it follows the same process as a normal user.**