Transactions by Location and IP

In this challenge, use the HTTP GET method to retrieve information from a database of Card Transactions records for users. Query https://jsonmock.hackerrank.com/api/transactions/search?userId=uid where uid is the value of user id passed to the function, to find all the records that belong to the user with id uid. The query response is paginated and can be further accessed by appending to the query string &page=num where num is the page number.

The query response from the API is a JSON response with the following five fields:

- page: the current page
- per page: he maximum numberlof results per page
- total: the total number of records in the search result
- total pages: the total number of pages which must be queried to get all the results
- data: an array of JSON objects that contains transaction records

The data field in the response contains a list of the transaction records, with each transaction record following the below-described schema :

- id: the unique ID of the record
- timestamp: the timestamp when the record was generated (In UTC milliseconds)
- userId: the user id of the user who performed the transaction
- userName: the user name of the user who performed the transaction
- txnType: the transaction type of the transaction. Allowed values are debit and credit
- amount: the transaction amount. Stored as a string with the currency structure and prefixed with the $ sign, e.g. "$2,273.95".
- location: object , the location description of the transaction
  - location.id: The id of the location where the transaction took place
  - location.address: The address of the location where the transaction took place
  - location.city: The city where the transaction took place
  - location.zipCode: The zip code of the location where the transaction took place
- ip: the IP address of the device which was used to perform the transaction

Given the user-id uid, the provided location id locationId, the ip address range netStart to netEnd, find and return the sum of the amount in the records that have the network prefix int the inclusive range

netStart to netEnd, belongs to the user uid and has been performed in the location with the id locationld. Note that the search is not case sensitive.

NOTE : netStart and netEnd refer to the first byte of the IPV4 address. So for example, to find the IP addresses in the range netStart and netEnd, take the first part of the IP address 172.187.11.212 which is 172 and match if it belongs between the range netStart and netEnd.

Function Description

Complete the function getTransactions in the editor below.

getTransactions has the following parameter(s):

int uid: the id by which record will be fetched

int locationld: the location id where the transaction took place

int netStart: the starting value (inclusive) of the network address for the device from which the transaction took place

int netEnd: the end value (inclusive) of the network address for the device from which the transaction took place Returns: int: the sum of amount for all transaction which matches the filter criteria rounded to the nearest integer

Input Format For Custom Testing

The first line contains an integer, uid.

The second line contains an integer, locationld.

The third line contains an integer, netStart.

The last line contains an integer, netEnd.

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN    Function
-----    --------
2    →    uid = 2
8    →    locationId = 8
5    →    netStart = 5
50    →    netEnd = 50
```

Sample Output

```
8446
```

Given *uid = 2* the query is *https://jsonmock.hackerrank.com/api/transactions/search?userId=2* and the response is:

```
[
 { id: 70,
   userId: 2,
   userName: 'Bob Martin',
   timestamp: 1550214607032,
   txnType: 'credit',
   amount: '$2,411.06',
   location:
    {
      id: 8,
      address: '389, Everest, Barwell Terrace',
      city: 'Murillo',
      zipCode: 66061
    },
   ip: '13.211.252.55'
}...
]
```

Apply the filter for *netStart* and *netEnd*, *netStart* is 5 and value of input *netEnd* is 50. The first record has an IP address 13.211.252.55. After formatting the IP address, the first byte is 13 which is between the range of *netStart* and *netEnd*. Thus this record passes the filter criteria. After applying this filter criteria for all the records with user ID 2 and location ID 8, there are 5 records. The sum of the amounts in these 5 records is 8446, the final output.