Create a contacts manager.

One should be able to view all contacts, add a contact, delete a contact and save all changes made to the data to a file.

You will need to create two classes: Contact and ContactList.

The Contact class should have the following properties:

- contact_id, an integer
- first_name, a string
- last_name, a string
- phone_number, a string

contact_id should be a read-only property. Therefore, use the @property decorator to make it return a private variable, _contact_id. There should be no setter.

The Contact class should also contain a display_data method, which prints out all the data (ID, First name, Last name, Phone number) in an orderly format.

The ContactList class should contain a private variable, _filename, which holds the name of the file all data will be saved to. Set it to "contacts.txt" in the constructor.

There should be two other properties:

- contacts, which would hold an array of Contact objects
- current_id, which would be the id of the next contact to be added. It should start from 1 and increment once a new contact is added.

current_id should be a read-only property. Use the @property decorator to make it return a private variable, _current_id. There should be no setter.

The constructor of the ContactList class is given below. Note that you would have to import os.path at the start of the code.
os.path.exists( ) was used to check if the file already exists.

Entries in contact.txt are saved as

```
id,first_name,last_name,number
```

with each entry on a new line.

arr = string.split(char) converts the string to an array by breaking it up with 'char'. The result is saved in arr.

```
def __init__(self):
        self._filename = "contacts.txt"
        self.contacts = []
        self._current_id = 1

        if os.path.exists(self._filename):
            f = open(self._filename, "r")
            for line in f:
                ln = line.strip("\n")
                arr = ln.split(",")
                new_contact = Contact(int(arr[0]), arr[1], arr[2], arr[3])
                self.contacts.append(new_contact)
            f.close()
            self._current_id = self.contacts[-1].contact_id + 1
```

Afterwards, still inside ContactList, implement the add_contact( ), remove_contact(contact_id), print_list( ) and save_changes( ) methods which adds a new contact, removes a contact with an ID, prints the contact list and saves all changes respectively.

Hint:

The main body of the program is given below:

```
my_phonebook = ContactList()

while True:
    print("Enter 1 to view phonebook")
    print("Enter 2 to add a new contact")
    print("Enter 3 to delete a contact")
    print("Enter 4 to save all changes")
```

```
    print("Enter anything else to quit")
    response = input("Your input: ")

    if response == "1":
        my_phonebook.print_list()

    elif response == "2":
        first_name = input("First name: ")
        last_name = input("Last name: ")
        phone_number = input("Phone number: ")
        new_contact = my_phonebook.add_contact(first_name, last_name,
phone_number)
        print("ID = " + str(new_contact.contact_id))
        print("First name = " + new_contact.first_name)
        print("Last name = " + new_contact.last_name)
        print("Phone number = " + new_contact.phone_number + "\n")

    elif response == "3":
        contact_id = int(input("Enter contact ID to delete: "))
        my_phonebook.remove_contact(contact_id)

    elif response == "4":
        my_phonebook.save_changes()

    else:
        print("\nGoodbye!")
        break
```