

Q1

function person(fname, lname, age, skills, dateOfBirth, address, married, profession) {

    this.fname = fname;

    this.lname = lname;

    this.age = age;

    this.skills = skills;

    this.dateOfBirth = dateOfBirth;

    this.address = address;

    this.married = married;

    this.profession = profession;

};

person1 = new person("Nikhil", "Goud", 22, ["C"], "24/10/1996", { city: "Hyderabad", pincode: "521185" }, false, "Sr. Analyst");

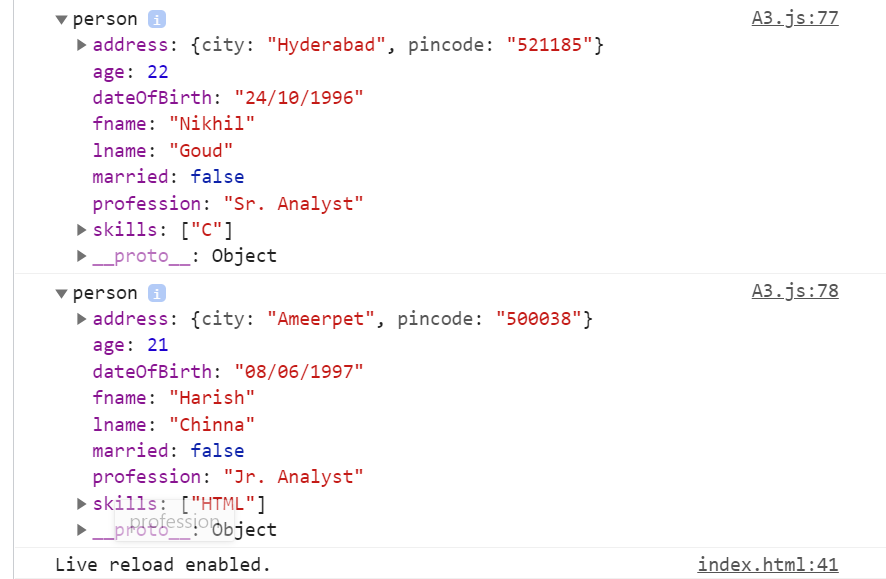
person2 = new person("Harish", "Chinna", 21, ["HTML"], "08/06/1997", { city: "Ameerpet", pincode: "500038" }, false, "Jr. Analyst");

print = function() {

    console.log(person1);

    console.log(person2);

}();



2.

amitabh = new person("Amitabh", "Bachchan", 22, ["C"], "24/10/1996", { city: "Hyderabad", pincode: "521185" }, false, "Sr. Analyst");

abhishek = {

    fname: "Abhishek",

    age: 21,

    skills: ["HTML"],

    dateOfBirth: "08/06/1997",

    married: false,

    profession: "Jr. Analyst"

};

abhishek = Object.assign(amitabh);

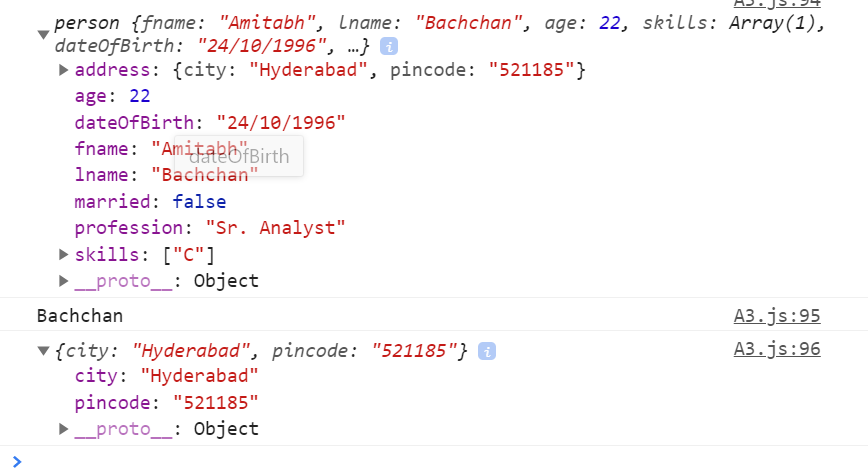
print = function () {

    console.log(amitabh);

    console.log(abhishek.lname);

    console.log(abhishek.address);

}();



3.

aaradhya = {

    ...amitabh,

    fname: "Aaradhya",

    age: 5,

    skills: ["crying"],

    dateOfBirth: "08/06/2016",

    married: false,

    profession: "school"

};

print = function () {

    console.log(aaradhya);

}();



4.

class BankAccount {

    constructor(accountNumber, accountHolderName, accountBalance) {

        this.accountNumber = accountNumber;

        this.accountHolderName = accountHolderName;

        this.accountBalance = accountBalance;

    }

};

class Savings extends BankAccount {

    constructor(accountNumber, accountHolderName, accountBalance, isSalary) {

        super(accountNumber, accountHolderName, accountBalance);

        this.isSalary = isSalary;

    }

}

class Current extends BankAccount {

    constructor(accountNumber, accountHolderName, accountBalance, odLimit) {

        super(accountNumber, accountHolderName, accountBalance);

        this.odLimit = odLimit;

    }

}

let savingsOne = new Savings("SA1", "Shiva", 2000, false);

let currentOne = new Current("CA1", "Boss", 5000, 1000);

Savings.prototype.withdraw = function (amount) {

    var newBalance = this.accountBalance - amount;

    if (newBalance < 0) {

        console.log("Insufficient balance");

    } else {

        this.accountBalance = newBalance;

        console.log("withdrawn amount is " + amount);

        return this.accountBalance;

    }

}

console.log(savingsOne.withdraw(1500));

Current.prototype.withdraw = function (amount) {

    var newBalance = this.accountBalance - amount;

    if (newBalance + this.odLimit < 0) {

        console.log("Insufficient balance");

    } else {

        this.accountBalance = newBalance;

        console.log("withdrawn amount is " + amount);

        return newBalance;

    }

}

console.log(currentOne.withdraw(500));

Savings.prototype.getCurrentBalance = function () {

    console.log("Savings account balance is " + this.accountBalance);

}

console.log(savingsOne.getCurrentBalance());

Current.prototype.getCurrentBalance = function () {

    console.log("current account balance is " + this.accountBalance);

}

console.log(currentOne.getCurrentBalance());

