```
//Import libraries
'use strict';
require('dotenv');
const _ = require('lodash');
const colors = require('colors');
const readlineSync = require('readline-sync');

const readlineInfo = require('./readlineInfo.js');
const commons = require('./commons.js');
const actions = require('./actions.js');

const fileName = 'contacts.txt';
const headers = ['Firstname', 'Lastname', 'Nickname', 'Phone', 'Email', 'Birthdate'
let contacts;

function main() {

  //MENU OPTION SELECTION
  let options = ['Add contact', 'Remove contact', 'Update contact', 'Contact list',

  // Reads text file and maps its element in an object array
  commons.readContactsFile(fileName)
    .then((data) => {
      return commons.createContactsList(data, headers);
    })
    .then((data) => {
      contacts = data;

      let index = readlineSync.keyInSelect(options, 'What do you want to do? ', {gu
      console.log(colors.bold('**' + options[index] + '**'));
      ++index;
      let functionName = actions.getActionFunction(index);
      functionName = 'start' + _.upperFirst(functionName);
      eval(functionName)()
    })
    .catch(console.log);
}

// TEST ENVIROMENT CONFIGURATION
if (!(process.env.NODE_ENV === 'test')) {
  main();
}

function startCreateContact() {
  readlineInfo.askContactInfo(headers)
    .then(answers => {
      let contact = {};
      _.each(answers, (a, i) => {
        contact[_.camelCase(headers[i])] = a;
      });
      actions.createContact(contact);
    });
}
```

```
// CALL ACTION METHODS
function startRemoveContact() {

  let contactName = readlineSync.question('Please write the name of contact that yo
  let found = commons.searchContacts(contacts, ['firstname', 'lastname'], contactNa

  if (!_.isEmpty(found)) {
    let contact = _.first(found);
    let sure = readlineSync.keyInYNStrict('Are you sure you want to delete ' + cont

    if (sure) {
      actions.removeContact(contacts, contact);
    }
  } else {
    console.log(colors.yellow('No contacts found to delete'));
  }
}

function startUpdateContact() {
  let contactName = readlineSync.question('Please write the name of contact that yo
  let found = commons.searchContacts(contacts, ['firstname', 'lastname'], contactNa

  if (!_.isEmpty(found)) {

    let contact = _.first(found);
    let chosen = readlineSync.keyInSelect(headers, 'What field do you want to updat

    let property = headers[chosen];
    let change = readlineSync.question('Please write the new ' + colors.bold(proper

    let update = actions.updateContact(contact, property, change);

    if (update.isUpdated) {
      contact = update.contact;
      commons.rewriteContactsFile(fileName, contacts)
        .then(() => {
          console.log('The contact ' + colors.bold(contact.firstname + ' ' + contac
        });
    } else {
      console.log(colors.bold.red('The contact has not been updated due to: ') + up
    }

  }
}

function startListContacts() {
  actions.listContacts(contacts);
}

function startSearchContact() {
  let searchIn = ['Complete name', 'Nickname', 'Phone', 'Email'];
  let chosen = readlineSync.keyInSelect(searchIn, 'Do you want to search by: ', {gu

  let property;
  switch (chosen) {
    case 0:
```

```
        property = ['firstname', 'lastname'];
        break;
      case 1:
      case 2:
      case 3:
        property = _.camelCase(searchIn[chosen]);
        break;
    }

    let value = readlineSync.question('Please write your ' + property.toString() + ':
    actions.searchContacts(contacts, property, value);
  }
```