به نام خدا

تمرین سوم درس برنامه‌نویسی پیشرفته

نیم‌سال اول ۱۳۹۹-۱۴۰۰

۰. فایل مربوط به توضیحات نحوه ارسال تمرین‌ها را که در مودل قرار دارد، مطالعه کنید.

۱. تمامی فایل‌های کد را به همراه فایل متنی که در قالب pdf است (مورد سوم را بخوانید) به صورت یک فایل آرشیو zip (zip != rar) که به قالب زیر نام‌گذاری شده است، بارگذاری نمایید.

StudentNumber_FirstName_LastName.zip

9031066_Ehsan_Edalat.zip

۲. در سوال‌هایی که ورودی و خروجی مطلوب آن‌ها مشخص شده است، برنامه‌ی شما به صورت ماشینی تصحیح می‌شود. بنابراین رعایت نحوه ورودی‌گرفتن و نمایش خروجی اهمیت بسیاری دارد. دقیقا همان‌طور که از شما خواسته شده است ورودی‌ها را خوانده و خروجی‌ها را تولید کنید.

۳. پاسخ سوالات تشریحی را به صورت تایپ‌شده و در قالب یک فایل pdf (برای کل تمرین) تحویل دهید.

۴. در صورت مشاهده هرگونه تقلبی، طبق موارد گفته شده در قوانین درس برخورد خواهد شد.

۵. در صورت وجود هرگونه ابهام می‌توانید از طریق ربات تلگرامی AP_Admin_bot با تدریس‌یاران در ارتباط باشید.

مهلت تحویل: تا جمعه ۲۵ مهر ۱۳۹۹ ساعت ۲۳:۵۵ شب

فهرست سوالات

صفحه

سوال اول..... ۳
سوال دوم..... ۳
سوال سوم..... ۴
سوال چهارم..... ۸

سوال اول

به سوالات زیر پاسخ دهید.

۱. توضیح دهید چرا نمی‌توان عضوهای یک `collection` را با استفاده از `foreach` حذف کرد؟

۲. متد `main()` از کدام `access modifier` باید استفاده کند؟ چرا؟

۳. قاعده‌ی `information hiding` را توضیح دهید.

سوال دوم

الف) درستی یا نادرستی عبارات زیر را همراه با توضیح مختصری تعیین کنید.

۱. اگر `constructor` کلاس `A` از نوع `private` تعریف شده باشد، فقط داخل خود کلاس `A` می‌توان از آن `instance` ساخت.

۲. فیلدهایی که `private` تعریف شده‌اند، داخل `package` خود قابل دسترسی‌اند.

۳. می‌توان تعداد محدودی آرگمان ورودی به `main()` داد.

ب) جاهای خالی را با عبارت مناسب پر کنید.

۱. با استفاده از _____، می‌توان به فیلدهای یک کلاس بدون ساخته شدن یک شی از آن کلاس دسترسی داشت.

۲. تابع `length()`، _____ و `size()`، _____ یک `ArrayList` را خروجی می‌دهند.

سوال سوم

(پیاده‌سازی)

در این سوال قصد داریم پیاده‌سازی دفترچه تلفن را انجام دهیم.

هدف اصلی این سوال آشنایی شما با برخی از مجموعه‌های زبان جاوا (مانند `arraylist`، `hashmap` و ...) می‌باشد. (انتظار می‌رود نحوه درست و مناسب ذخیره سازی را تشخیص داده و پیاده‌سازی نمایید)

کلاس `Info`: از این کلاس جهت ذخیره سازی اطلاعات یک مخاطب استفاده می‌گردد.

*** فیلدها:**

- **`workInfo`**: اطلاعات شغل مخاطب که به صورت رشته ذخیره می‌گردد.

- **`groups`**: هر مخاطب می‌تواند عضو تعدادی گروه باشد (ممکن است عضو گروهی نباشد) که هر گروه به صورت یک رشته بیان می‌شود.

- **`emails`**: دقیقا مشابه گروه می‌باشد.

- **`phones`**: هر مخاطب باید حداقل یک شماره تماس داشته باشد که به صورت رشته دقیقا ۱۲ حرفی ذخیره می‌گردد.

*** توابع:**

- `public void printInfo()`

در تابع مذکور، شما باید اطلاعات مخاطب (تمامی فیلدها) را در قالب مناسب نمایش دهید.

کلاس `PhoneBook`: در این کلاس مخاطبان به همراه اطلاعاتشان ذخیره می‌گردد.

*** فیلدها:**

- **`contacts`**: ذخیره هر مخاطب در عنوان یک `map` متشکل از نام کامل و اطلاعات مخاطب.

*** توابع:**

- `public boolean addContact(String name, Info info)`

باید مخاطبی را به لیست مخاطبین اضافه کند. اگر مخاطب در لیست موجود باشد اطلاعتش به روز می‌شود و خروجی تابع `true` می‌شود.

- `public Info findContact(String name)`

باید مخاطبی با نام داده شده را برگرداند، اگر مخاطب موجود نبود، خروجی تابع `Null` خواهد بود.

- `public boolean deleteContact(String name)`

باید مخاطبی با نام داده شده را از لیست مخاطبین حذف کنید، اگر مخاطب در لیست موجود نبود باید خروجی تابع `false` باشد.

- `public HashMap<string, Info> findContacts(String group)`

شما باید تمامی مخاطبان عضو گروه وارد شده را خروجی دهید.

- `public void printContacts()`

این تابع وظیفه نمایش لیست مخاطبان را به صورت مناسب بر عهده دارد.

- `public void printContacts(String group)`

این تابع وظیفه نمایش لیست مخاطبان که دارای گروهی خاص هستند را به صورت مناسب بر عهده دارد. (`overload`)

پس از پیاده سازی کلاس‌ها و متدهای بیان شده و همچنین `getter` و `setter` های مورد نیاز، نوبت به پیاده سازی `main` و استفاده از موارد ایجاد شده می‌رسد.

ابتدا توابع مقابل را پیاده سازی نمایید:

حال در `main` حلقه‌ای پیاده سازی کنید که به صورت زیر ورودی گرفته و خروجی مورد نظر را تولید کند:

1. `contacts -a <contact name>`
2. `contacts –r <contact name>`
3. `show –g <group name>`
4. `show –c <contact name>`

```
5. show
6. exit
```

۱. به منظور اضافه کردن مخاطب – در ادامه مثالی از این فرآیند خواهید دید.

۲. به منظور حذف یک مخاطب – خروجی نمایش‌داده شده برای حالتی که مخاطب وجود داشته باشد، Ok و برای حالتی که وجود نداشته باشد، Not found است.

۳. به منظور نمایش دادن اطلاعات مخاطبان یک گروه

۴. به منظور نمایش اطلاعات یک مخاطب

۵. نمایش نام تمام مخاطبان

۶. خروج از برنامه

دقت کنید که در این برنامه، نوع خروجی‌ای که نمایش می‌دهید باید قابل خواندن باشـد (خروجی فرمت مناسبی برای خواندن راحت داشته باشد) و فرمت کلی‌ای برای سوال داده نشده است.

پیاده‌سازی توابع گفته شده الزامی است اما شما می‌توانید توابع اضافه‌ای نیز تعریف کنید و استفاده کنید.

برای مثال به خروجی‌های زیر توجه فرمایید:

مثال برای تابع Show:

```
User

Professor Bakhshi : Teacher

Phones:

1 -> 989123456789

Groups:

1 -> teacher

2 -> aut

emails:

-----------------------------------
```

مثال اضافه کردن مخاطب:

Input: contacts –a Professor Bakhshi

Output: “Please enter work info: ”

Input: Teacher

Output: “Please enter group 1 or enter to finish: ”

Input: teacher

Output: “Please enter group 2 or enter to finish: ”

Input: aut

Output: “Please enter group 3 or enter to finish: ”

Input:

Output: “Please enter group 3 or enter to finish: ”

Input:

Output: “Please enter email 1 or enter to finish: ”

Input:

Output: “Please enter phone number 1: ”

Input: 989123456789

Output: “Please enter phone number 2 or enter to finish: ”

Input:

Output: “Contact Saved!”

سوال چهارم

(پیاده‌سازی)

در این سوال قصد داریم رزواسیون یک هتل را شبیه سازی کنیم.

کلاس Room: از این کلاس برای رزرو کردن اتاق و مشخصات اتاق‌ها استفاده میکنیم.

*** فیلدها:**

- **number**: شماره‌ی اتاق
- **view**: منظره‌ی اتاق (تایپ این متغیر باید از جنس `enum` باشد)

```
enum View {
        SEA,
        FOREST,
        BULDING
}
```

- **type**: نوع اتاق (دارای تایپ `Type` که به صورت `enum` با مقادیر`Single`، `Double`، `Triple`، `Quad`، `Queen` و `King` تعریف می‌شود)
- **date**: بازه‌ای که در آن اتاق رزرو است (تایپ این متغیر در ادامه توضیح داده می‌شود)
- **price**: هزینه روزانه اقامت

*** توابع:**

- `public void printInfo()`

در تابع مذکور شما باید اطلاعات اتاق (تمامی فیلدها) را در قالب مناسب نمایش دهید.

- `public void reserve(Date date)`

در تابع مذکور شما باید بتوانید اتاق را رزرو کنید.

- `public boolean isReserved(Date date)`

در تابع مذکور شما باید بفهمید که در بازه ورودی اتاق رزرو است یا خیر.

- `public View getView()`

در متد مذکور شما میتوانید منظره اتاق را بفهمید.

- `public Type getType()`

در متد مذکور شما میتوانید نوع اتاق را بفهمید.

- `public int getNumber()`

در متد مذکور شما میتوانید شماره اتاق را بفهمید.

کلاس `Hotel` :از این کلاس برای سازماندهی و تعامل هتل و مسافران استفاده میشود.

*** فیلدها:**

- **`rooms`**: اتاق های هتل
- **`guests`**: لیست مهمان‌ها

*** توابع:**

- `public void roomList()`

در تابع مذکور شما باید اطلاعات اتاق‌ها را در قالب مناسب نمایش دهید.

- `public ? getRoomList()`

در تابع مذکور شما به اتاق‌های هتل دسترسی دارید (توجه کنید که `return type` مناسب را خودتان باید تشخیص دهید).

- `public ? getGuestList()`

در تابع مذکور شما میتوانید لیست مهمانان هتل دسترسی داشته باشید(توجه کنید که `return type` مناسب را خودتان باید تشخیص دهید).

- `public void reserve(Guest guest, Date date, ? type, ? view)`

در تابع مذکور شما باید با ورودی گرفتن مسافر، بازه اقامت، نوع اتاق و منظره اتاق، اتاق مناسبی را رزرو کنید.(علامت سوال به علت تعیین نوع داده‌ها توسط شما قرار داده شده است.

- `public void checkIn(Guest guest)`

در تابع مذکور شما می‌توانید یک مهمان را به هتل اضافه کنید.

- `public int checkOut(Guest guest)`
  در متد مذکور یک مهمان را از لیست هتل حذف کنید و هزینه اقامت را خروجی دهید.

کلاس Guest: از این کلاس برای سازماندهی مسافران و انجام کارهای آنها استفاده میشود.

❋ فیلدها:

- **name**: نام مسافر
- **room**: اتاقی که در آن مستقر میباشد.

❋ توابع:

- `public void setName (String name)`
  در متد مذکور شما میتوانید نام مسافر را تعیین کنید.
- `public String getName ()`
  در متد مذکور شما میتوانید به نام مسافر دسترسی داشته باشید.
- `public void bookHotel (Room room)`
  در متد مذکور شما باید بتوانید یک اتاق را رزرو کنید.

کلاس Date: از این کلاس برای ایجاد تاریخ استفاده میشود.

❋ فیلدها:

- **daysToStay**: تعداد روزهای اقامت
- **start**: تاریخ شروع اقامت.
- **finish**: تاریخ ترک هتل.

❋ توابع:

- `public void setStartDay (String start)`

در تابع مذکور شما میتوانید شروع اقامت را تعیین کنید. برای مثال ۱۳۹۹۰۱۰۵ به معنای ۵ فرودین سال ۱۳۹۹ است.

- `public String getStartDate ()`
  در تابع مذکور شما میتوانید به شروع اقامت دسترسی داشته باشید.
- `public void setFinishDay (String finish)`
  در تابع مذکور شما میتوانید پایان اقامت را تعیین کنید.
- `public String getFinishDate ()`
  در تابع مذکور شما میتوانید به پایان اقامت دسترسی داشته باشید.
- `public void setDaysToStay (String finish)`
  در تابع مذکور شما میتوانید تعداد روزهای اقامت خود را تعیین کنید.
- `public String getDaysToStay ()`
  در تابع مذکور شما میتوانید به تعداد روزهای اقامت خود دسترسی داشته باشید.

منوی مناسبی را در قالب کلاس `Main` پیاده‌سازی کرده و در آن از قابلیت‌های کلاس هتل برای ایجاد سامانه استفاده نمایید.

توجه داشته باشید که پیاده سازی این برنامه نباید به طوری باشد که بعد از هر عملیات از برنامه خارج شویم و امکان این را داشته باشیم که مهمان‌های متعددی از این برنامه پشت سرهم بتوانند استفاده کنند .

همچنین توابع مطرح شده، صرفا توابع پیشنهادی می‌باشند و می‌توانید برحسب نیاز توابع بیشتری تعریف نمایید (عملکرد سامانه نباید تغییر کند).