کواک

- نام طراح: ایلیا آخوندی
- سطح سوال: ساده

شاید با اصطلاح **DuckTyping** در برنامه نویسی آشنا باشید. duckTyping یک typeSystem می باشد که در زبان های به اصطلاح dynamic استفاده می شود. (مانند Python و Php و Perl و ...). از این روش زمانی استفاده می‌شود که که نوع یا کلاس یک شی از متدی که تعریف می کند اهمیت کمتری دارد. با استفاده از Duck Typing، ما اصلاً انواع داده را بررسی نمی کنیم. در عوض، ما وجود یک متد یا ویژگی مشخص را بررسی می کنیم. اصطلاح معروف برای ducktyping:

> If it looks like a duck and quacks like a duck, it’s a duck

(اطلاعات بیشتر)

با وجود این که فردی فقط و فقط برای شوخی و به روش های غیر ممکنی در جاوا duckTyping را ممکن کرده است(اینجا را کلیک کنید) ولی در مجموع انجام اینکار در جاوا غیر ممکن است. اما بهنام که بعد از آشنایی کمی با شی گرایی و ارث بری در جاوا میخواهد با استفاده از امکانات شی گرایی جاوا به گونه ای DuckTyping را به صورت ابتدایی پیاده سازی کند.

بهنام باید کاری کند تا با اجرای کد زیر:

```
public static void main(String[] args) {
    DuckObject object = new Dog();
    DuckObject object1 = new Duck();
    DuckObject object2 = new Human();
    DuckObject[] objects = new DuckObject[3];
    objects[0] = object;
    objects[1] = object1;
    objects[2] = object2;
    for (DuckObject myObjects : objects) {
        myObjects.doSth();

```

```
12        }
    }
```

خط های زیر چاپ شوند:

```
Hop Hop!
The Dog walked

quack quack!
The Duck walked

Hello!
The Human walked
```

(دقت داشته باشید که بعد از هر دو خط یک enter چاپ می شود).

راهنمایی

سعی کنید تابع doSth را به گونه ای پیاده سازی کنید که با استفاده از صدا زدن تابع های مورد نظر که اسم یکسانی دارند ولی رفتار متفاوت عبارات چاپ شوند.

آنچه که باید آپلود کنید

کلاس های Duck , Human , Dog و DuckObject را در فایل به نام Main.java بدون هیچ پکیج بندی خاصی آپلود کنید.

انتخاب واحد

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- نام طراح: محمدسعید زارع
- سطح سوال: نسبتا ساده

سعید که فردا انتخاب واحد دارد حوصله ندارد ظرفیت کلاس ها، تعداد ثبتنامی ها و تداخل ها را بررسی کند برای همین به دنبال راهی است که این مشکل را حل کند. یک سیستم انتخاب واحد بنویسید که سعید کم حوصله بتواند با کمک آن انتخاب واحد خوبی را پشت سر بگذارد.

برای بهتر درک کردن سوال، یکبار آن را تا انتها بخوانید و سپس شروع به کد زدن کنید.

اینام Day

در این `enum` روزهای هفته نوشته شده است.

```
MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY
```

کلاس Date

کلاس `Date` دارای ویژگی های زیر میباشد. `dayOfWeek` روز هفته، `startTime` زمان شروع کلاس و `endTime` زمان پایان کلاس

```
1 | Day dayOfWeek;
2 | double startTime;
3 | double endTime;
```

کلاس `Day` دارای کانستراکتور زیر میباشد.

```java
public Date(Day dayOfWeek, double startTime, double endTime){
    // TODO
}
```

کلاس Course

کلاس Course دارای ویژگی های زیر میباشد. courseId شناسه‌ی عددی است که با استفاده از آن می‌توان به آن درس دسترسی داشت، اولین درسی که ساخته می‌شود شناسه‌ی عددی ۱، دومین درس، شناسه‌ی عددی ۲ و i-امین درس، شناسه‌ی عددی i دارد. name نام، capacity ظرفیت درس، students آرایه ای از دانشجویان ثبتنامی، noStudents تعداد دانشجویان ثبتنامی، unit واحد و classTimes زمان های برگزاری کلاس میباشد.

```java
private int courseId;
private String name;
private int capacity;
private Student[] students;
private int noStudents;
private int unit;
private Date[] classTimes;
```

کلاس Course دارای کانستراکتور زیر میباشد.

```java
public Course(String name, int capacity, int unit, Date[] dates){
    // TODO
}
```

این کلاس دارای متدهای زیر میباشد.

- متد addStudent

این متد یک دانشجو را دریافت میکند و به لیست دانشجوهای این درس اضافه میکند.

```java
public void addStudent(Student student){
  // TODO
```

```
}

```

- متد deleteStudent

این متد یک دانشجو دریافت میکند و از لیست دانشجوهای درس در صورت وجود حذف میکند.

```
public void deleteStudent(Student student){
  // TODO
}
```

- متد isFull

این متد در صورتی که این درس پر شده باشد true برمیگرداند در غیر اینصورت false برمیگرداند.

```
public boolean isFull() {
  // TODO
}
```

- متد changeCapacity

این متد یک عدد دریافت میکند که ظرفیت جدید کلاس است، در صورتی که دانشجوهای ثبتنامی بیشتر از این عدد باشند و منجر به حذف شدن یک یا چند دانشجو از کلاس شود این متد کاری انجام نمیدهد در غیر اینصورت ظرفیت کلاس تغییر میکند.

```
public void changeCapacity(int capacity){
  // TODO
}
```

- متد toString این کلاس را به گونه ای پیاده سازی کنید که اگر درسی با آیدی 4، نام ALG، تعداد واحد 3 و 5 نفر جای خالی برای ثبتنام وجود داشته باشد، رشته زیر برگردانده شود.

```
{id : 4, name : AlG, emptyCapacity : 5, unit : 3}
```

کلاس Student

کلاس Student دارای ویژگی های زیر میباشد. stuId شماره دانشجویی هر فرد هست که نفر اول شماره دانشجویی 400243001، نفر دوم 400243002 و ... (**تضمین میشود تعداد دانشجویان این ورودی کمتر از 99 تاست.**)، name نام، grade معدل، courses درس های برداشته شده توسط دانشجو و noCourses تعداد این درس هاست.

```
private String stuId;
private String name;
private double grade;
private Course[] courses;
private int noCourses;
```

این کلاس دارای کانستراکتور زیر میباشد.

```
public Student(String name, double grade){
    // TODO
}
```

این کلاس دارای متدهای زیر میباشد.

- متد addCourse

این متد یک درس دریافت میکند، اگر زمان برگزاری این درس با درس دیگری تداخل داشت پیغام conflict with another course ، اگر این درس توسط دانشجو اخذ شده بود پیغام this course already added to your courses list و در صورت پر بودن کلاس پیغام this class capacity is full برگردانده میشود. دانشجو نمیتواند بیشتر از سقف واحد مجاز، درس بردارد در اینصورت پیغام you have reached the maximum allowance فرستاده میشود. (**دانشجویان با معدل بالای 17، 24 واحد، با معدل 12 تا 17، 20 واحد و بقیه دانشجویان تنها مجاز به اخذ 12 واحد هستند.**) *نکته: اگر چند تا از اتفاق های بالا همزمان اتفاق افتاد همه پیام ها با جداکننده کاما و به ترتیب گفته شده برگردانده میشود.* در صورتی که هیچ کدام از اتفاق های بالا

رخ ندهد درس با موفقیت اخذ میشود و پیغام `this course added to the list successfully` برگردانده میشود.

```
public String addCourse(Course course){
  // TODO
}
```

- متد deleteCourse

این متد یک درس دریافت میکند و آن را از لیست درس های اخذ شده حذف میکند. اگر این درس وجود نداشت پیغام `course doesn't exist` و اگر با موفقیت حذف شد پیغام `course deleted` برگردانده میشود.

```
public String deleteCourse(Course course){
  // TODO
}
```

- متد printCourseTable

این متد جدول هفتگی دانشجو را به فرمت خواسته شده چاپ میکند.

```

          09-11 11-13 13-15 15-17 17-19
Saturday  ***** ***** ***** ***** *****
Sunday    AlG   ***** DS    ***** *****
Monday    DB    ***** ***** ***** *****
Tuesday   AlG   ***** ***** ***** *****
Wednesday ***** ***** ***** ***** *****
Thursday  ***** ***** ***** ***** *****
Friday    ***** ***** ***** ***** *****
```

- **هر خانه از جدول 5 کاراکتر جا دارد پس اگر یک درس نامی بیشتر از 5 کاراکتر داشت فقط 5 کاراکتر اولش را در جدول قرار دهید و اگر کمتر از 5 کاراکتر بود اسپیس قرار دهید.**
- **در صورتی که در خانه ای از جدول درسی وجود نداشت، بجای آن پنج کاراکتر * قرار دهید.**

- تضمین میشود زمان برگزاری درس هایی که داده میشوند در یکی از بازه های بالا قرار بگیرد.(برای مثال کلاس میتواند 9.5 تا 11 برگزار شود ولی 9.5 تا 11.5 نه!)
- متد toString

این متد را به گونه ای پیاده سازی کنید که برای فردی با شماره دانشجویی 400243099 ، نام saeed و معدل 16.51 رشته زیر برگردانده شود.

```
{stuId : 400243099, name : saeed, grade : 16.51}
```

اگر کد شما به درستی کار کند با دادن ورودی زیر، خروجی زیر نیز باید چاپ شود.

ورودی

```
public static void main(String[] args) {
        Student stu1 = new Student("saeed", 13);
        Date d1 = new Date(Day.SUNDAY, 11, 13);
        Date d2 = new Date(Day.MONDAY, 11, 13);

        Date d3 = new Date(Day.SUNDAY, 10, 11);
        Date d11 = new Date(Day.SATURDAY, 10, 11);
        Date d4 = new Date(Day.MONDAY, 9, 11);

        Date d5 = new Date(Day.SATURDAY, 9, 11);
        Date d6 = new Date(Day.THURSDAY, 17, 19);

        Date d7 = new Date(Day.TUESDAY, 14, 15);
        Date d8 = new Date(Day.THURSDAY, 13, 15);

        Date d9 = new Date(Day.MONDAY, 15.5, 16.5);
        Date d10 = new Date(Day.WEDNESDAY, 16.5, 17);

        Course c1 = new Course("DB", 5, 3, new Date[]{d1, d2});
        Course c2 = new Course( "DataStructure", 5, 3, new Date[]{d4, d5});
        Course c3 = new Course( "AP ta", 0, 3, new Date[]{d6});
        Course c4 = new Course( "DSD", 5, 3, new Date[]{d2});
        Course c5 = new Course( "Algorithm", 5, 3, new Date[]{d3, d11});
        Course c6 = new Course( "DS", 5, 3, new Date[]{d7, d8});
```

```
        Course c7 = new Course("Math", 10, 2, new Date[]{d9, d10});
        System.out.println(stu1.addCourse(c1));
        System.out.println(stu1.addCourse(c2));
        System.out.println(stu1.addCourse(c3));
        c3.changeCapacity(1);
        System.out.println(stu1.addCourse(c3));
        System.out.println(stu1.addCourse(c4));
        System.out.println(stu1.addCourse(c5));
        System.out.println(stu1.addCourse(c6));
        System.out.println(stu1.addCourse(c7));
        System.out.println(stu1.addCourse(c7));
        System.out.println("---");
        System.out.println(stu1.printCourseTable());
        System.out.println("---");
        System.out.println(stu1.deleteCourse(c5));
        System.out.println(stu1.deleteCourse(c3));
        System.out.println("---");
        System.out.println(stu1.printCourseTable());
        System.out.println("---");
        System.out.println(stu1);
        System.out.println(c5);
    }

```

خروجی

```
this course added to the list Successfully
this course added to the list Successfully
this class capacity is full
this course added to the list Successfully
conflict with another course
conflict with another course
this course added to the list Successfully
this course added to the list Successfully
conflict with another course, this course already added to your courses list
---
          09-11 11-13 13-15 15-17 17-19
Saturday  DataS ***** ***** ***** *****
Sunday    ***** DB    ***** ***** *****
Monday    DataS DB    ***** Math  *****
```

```
Tuesday   ***** ***** DS    ***** *****
Wednesday ***** ***** ***** Math  *****
Thursday  ***** ***** DS    ***** AP ta
Friday    ***** ***** ***** ***** *****
---
course doesn't exist
course deleted
---
          09-11 11-13 13-15 15-17 17-19
Saturday  DataS ***** ***** ***** *****
Sunday    ***** DB    ***** ***** *****
Monday    DataS DB    ***** Math  *****
Tuesday   ***** ***** DS    ***** *****
Wednesday ***** ***** ***** Math  *****
Thursday  ***** ***** DS    ***** *****
Friday    ***** ***** ***** ***** *****
---
{stuId 400243001, name : saeed, grade : 13.0}
{id : 5, name : Algorithm, emptyCapacity : 5, unit : 3}
```

نکات

- در صورت نیاز میتوانید متدها و فیلدهای جدیدی به کلاس ها اضافه کنید.

آنچه باید آپلود کنید

فایل Main.java حاوی تمام کلاس های گفته شده را بدون هیچ پکیج بندی آپلود کنید.

دیکشنری

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- نام طراح: محمدسعید زارع
- سطح سوال: ساده

از انجایی که گوگل ترنسلیت فیلتر شده و سعید میخواهد مقاله ای در حوزه وب ترجمه کند تصمیم میگیرد، یک دیکشنری انگلیسی به فارسی برای خودش پیاده سازی کند ولی چون دانش این کار را ندارد از شما میخواهد آن را پیاده سازی کنید.

ورودی و خروجی نمونه برای درک بهتر سوال آمده اند پس اگر ابهامی در صورت سوال داشتید به آن مراجعه کنید.

کلاس Pair

کلاس `Pair` برای نگه داری لغت یا url و معنی آن پیاده سازی شده است و حاوی ویژگی های زیر میباشد. `key` لغت و `value` معنی آن است.

```
private String key;
private String value;
```

کلاس `Pair` دارای کانستراکتور زیر میباشد.

```
public Pair(String key, String value){
    // TODO
}
```

کلاس Dictionary

کلاس Dictionary کلیه لغات را نگه داری میکند و حاوی ویژگی های زیر میباشد. name نام دیکشنری، size حداکثر گنجایش دیکشنری و pairs لیستی از لغت ها و معنی هایشان

```
private String name;
private int size;
private Pair[] pairs;
```

کلاس Dictionary دارای کانستراکتور زیر میباشد.

```
public Dictionary(String name, int size){
    // TODO
}
```

متدها

- متد containsKey

این متد یک لغت یا url دریافت میکند و با توجه به وجود آن در دیکشنری یک مقدار boolean برمیگرداند.

```
public boolean containsKey(String key){
  // TODO
}
```

- متد put

این متد لغت و معنی آن را دریافت میکند و در صورتی که آن لغت در دیکشنری وجود نداشته باشد و دیکشنری پر نباشد آن را به دیکشنری اضافه میکند.

```
public boolean put(String key, String value){
  // TODO
}
```

- متد `putUrl`

متد `putUrl` مقدار url و معنی آن را دریافت میکند. این متد همانند متد `put` است با این ویژگی که ورودی این تابع حتما باید url باشد.

شروط url بودن: (prefix.domain.postfix)

- حتما باید با یکی از پیشوند های `https://` ، `www.` ، `http://` شروع شود. (prefix)
- حتما باید با یکی از پسوند های `org` ، `com` ، `ir` تمام شود. (postfix)
- `domain` باید با حروف کوچک یا بزرگ انگلیسی شروع میشود.
- `domain` حداقل 3 کاراکتر و حداکثر 20 کاراکتر دارد که این کاراکتر ها میتوانند حروف کوچک و بزرگ انگلیسی، عدد و نقطه `.` باشند.

```
public boolean putUrl(String url, String value){
  // TODO
}
```

- متد `isEmpty`

این متد خالی بودن دیکشنری را بررسی میکند و در صورت خالی بودن مقدار true برمیگرداند.

```
public boolean isEmpty(){
  // TODO
}
```

- متد `replace`

این تابع معنی یک لغت را در صورت وجود عوض میکند. اگر این لغت وجود نداشت کاری انجام نمیشود.

```
public void replace(String key, String value){
  // TODO
}
```

- متد `get`

این متد لغت را دریافت میکند و معنی را برمیگرداند. در صورت وجود نداشتن لغت مقدار `no value!` برگردانده میشود.

```
public String get(String key){
  // TODO
}
```

- متد `remove`

این متد یک لغت دریافت میکند و درصورت وجود آن را حذف میکند.

```
public void remove(String key){
  // TODO
}
```

- متد `sort`

این متد لغت ها و معنی هایشان را بر اساس ترتیب لغتنامه ای به صورت صعودی مرتب میکند. (به ورودی نمونه دقت کنید.) **استفاده از فانکشن های built-in جاوا ممنوع است.**

```
public void sort(){
  // TODO
}
```

- متد `print`

این متد لغت و معنی آنها را به فرمت خاصی چاپ میکند. برای مثال اگر داخل دیکشنری لغت key1، key2 و key3 با معانی value1، value2 و value3 موجود باشند خروجی به این صورت است.

```
{
key1 -> value1
key2 -> value2
```

```
key3 -> value3
}
```

```
public String print(){
  // TODO
}
```

- متد `size`

این متد تعداد کلمات داخل دیکشنری را برمیگرداند.

```
public int size(){
  // TODO
}
```

اگر کد شما به درستی کار کند با دادن ورودی زیر، خروجی زیر نیز باید چاپ شود.

ورودی

```
public static void main(String[] args) {
Dictionary dictionary = new Dictionary("dictionary", 10);
dictionary.put("summit", "ghole"); // true
dictionary.put("example", "mesal"); //true
dictionary.put("ablaze", "soozandan"); // true
dictionary.put("reason", "dalil"); //true
dictionary.put("slight", "ka m"); //true
dictionary.putUrl("https://www.varzesh3.com", "varzesh 3"); //true
dictionary.putUrl("https://1quera.org", "quera"); //false
dictionary.putUrl("ww.fhsdakl.ir", "no!"); //false
dictionary.put("dart", "==>"); //true
dictionary.put("cut", "boridan"); //true
dictionary.remove("dart");
dictionary.remove("cut");
dictionary.remove("summit");
dictionary.replace("slight", "kam");
dictionary.put("slightly", "kami");
if (dictionary.containsKey("slight")){
```

```
System.out.println(dictionary.get("slight"));
}
System.out.println(dictionary.get("key")); // no value!
System.out.println(dictionary.get("https://www.varzesh3.com"));
dictionary.sort();
System.out.println(dictionary.size());
System.out.println(dictionary.print());
}
```

خروجی

```
kam
no value!
varzesh 3
6
{
ablaze -> soozandan
example -> mesal
https://www.varzesh3.com -> varzesh 3
reason -> dalil
slight -> kam
slightly -> kami
}
```

نکات

- در صورت نیاز میتوانید متدها و فیلدهای جدیدی به کلاس ها اضافه کنید.
- برای سورت لغت ها خیلی سخت نگیرید :)

آنچه باید آپلود کنید

فایل Main.java حاوی تمام کلاس های گفته شده را بدون هیچ پکیج بندی آپلود کنید.

قلی(git)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: یزدان جاهدی
- سطح: ساده

دستورات پایه‌ای گیت

قلی بعد از یاد گرفتن برنامه نویسی، قصد دارد تا اولین پروژه بزرگ خود را انجام دهد. برای همین در ابتدا چند فایل اولیه پروژه خود را می سازد تا در ادامه تکمیلشان کند. سارختار فایل های پروژه قلی به این شکل است:

```
project
├── index.html
├── js
│   └── a.js
├── style.css
└── temp
```

اما متاسفانه قلی `git` را بلد نیست و از شما برای استفاده از گیت در پروژه‌اش کمک گرفته است.

گیت از قبل، روی سیستم قلی نصب شده است اما کانفیگ های اولیه انجام نشده است.

۱و۲- شما در ابتدا باید که به صورت **سراسری** اسم و ایمیل او را وارد کنید. (اسم او را `goli` و ایمیل او را `goli@gmail.com` وارد کنید)

بعد از این کار، ترمینالی را در پوشه پروژه باز میکنید و باید دستوراتی را وارد کنید:

۳- در ابتدا `git .` را تشکیل میدهید.

*۴-** سپس همه فایل های موجود در پروژه را به staging area، *اضافه* می کنید تا در ادامه تغییرات اولیه را بتوانید ذخیره کنید.

▼ راهنمایی

برای دستور چهارم، نباید نام فایل ها را تک تک بنویسید یا اضافه کنید

در همین زمان قلی یاد فایل `temp` موجود در پوشه می افتد که نمیخواهد ذخیره شود. پس:

۵- با دستوری فایل temp را از staging area خارج کنید.

۶- حال همه تغییرات قبلی فایل ها مد نظر را ذخیره (کامیت) کنید. متن کامیت را `initiall commit` بگذارید.

قلی که متوجه اشتباه تایپی شما در متن کامیت قبلی میشود، از شما میخواهد تا پیام کامیت قبلی را درست کنید:

۷- پیام کامیت قبل را به `initial commit` تغییر دهید. **دقت کنید** که با استفاده از یک دستور و در یک خط باید این اتفاق بیفتد و نیاز به کار دیگری نباشد.

۸- برای اطیمنان بیشتر از مراحل انجام شده تا کنون، با دستوری وضعیت کنونی را **بصورت خلاصه** مشاهده کنید.

▼ راهنمایی

در حال حاضر، حالت فایل temp باید به صورت `??` باشد.

قلی دست به اعمال یک سری تغییرات روی فایل `a.js` میزند و همچنین متوجه نام‌گذاری بد این فایل میشود! او از شما میخواهد که نام این فایل را به `myfile.js` تغییر داده و تغییرات اعمال شده را ذخیره (کامیت) کنید:

۹- نام فایل `a.js` را به `myfile.js` تغییر بدهید.

۱۰- تغییر نام فایل را با متن `js file is renamed` کامیت کنید.

۱۱- فایل `myfile.js` را به staging area اضافه کنید.

۱۲- تغییرات را با پیام `js file is changed` کامیت کنید.

بعد از این قلی متوجه میشود که تغییرات ایجاد کرده شده در فایل js، خوب و مفید نبوده اند و میخواهد که به حالت قبل این تغییرات برگردیم! شما به عنوان کار بلد، این کار را برایش انجام دهید!

۱۳- لیست کامیت ها را به صورت خلاصه و **در یک خط** مشاهده کنید. فرض کنید که commit hash، کامیت مورد نظر ما این چنین است: `07a0ce0`

▼ راهنمایی

خروجی دستور بالا، باید چنین شکلی داشته باشد:

```
72b9744 (HEAD -> master) js file is changed
07a0ce0 js file is renamed
ec0d208 initial commit
```

۱۴- پروژه را به کامیت گفته شده **ریست** کنید.

▼ راهنمایی

باید دقیقا به همان کامیت برگردیم و نباید دیگر تغییرات اعمال شده در فایل js مشاهده شود.

قلی بابت این همه زحمتی که به شما داده از شما تشکر میکند و تصمیم میگرید که خود به یادگیری گیت بپردازد تا دیگر مزاحم شما نشود!!

نکات تکمیلی:

- پیشنهاد میکنم حتما مراحل گفته شده در صورت سوال را امتحان و بررسی کنید و صرفا به نوشتن جواب ها کفایت نکنید.
- همه دستورات وارد شده باید مخصوص گیت باشند. پس همه دستورات با کلمه git شروع خواهند شد.
- شما برای پاسخ، باید فایلی به نام answer.txt ایجاد کرده و هر یک از دستورات را دقیقا به ترتیب نوشته شده در سوال بنویسید و ارسال کنید.
- پس فایل ارسالی شما **دقیقا دارای ۱۴ خط** خواهد بود.
- اگر جواب یک دستور را نمیدانید، آن خط را خالی بگذارید و به خط بعدی بروید.

- اگر همه نکات گفته شده در سوال را رعایت کردید و به جواب نهایی خود اطمینان دارید ولی نمره کامل را نگرفتید، میتوانید به من پیام دهید!

همکاری(git)

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت
- طراح: یزدان جاهدی
- سطح: ساده

برنچ، ریموت

بعد از رهایی از دست قلی، شما تصمیم به همکاری در یک پروژه متن باز کرده اید. وظیفه شما به عنوان همکار در پروژه، توسعه قسمت فرانت-اند است. شما برای شروع کردن به کار، در ابتدا یک پوشه همراه با گیت برای خود ایجاد کرده اید و قصد دارید که فایل های مربوط به پروژه را در آن نگه دارید.

۱- در ابتدا آدرس منبع ریموت را با نام `origin` به گیت خود معرفی کنید. آدرس منبع این است:

```
https://github.com/someusername/the-project
```

۲- حال فایل های منبع ریموت را به سیستم خود اضافه کنید (**بکشید!**).

فایل های اضافه شده به این شکل هستند (که به طرز اتفاقی، بسیار شبیه پروژه قلی است!!):

```
.
├── index.html
├── js
│   └── myfile.js
├── style.css
└── temp
```

*۳-** *همانطور که گفته شد، شما در این پروژه تازه وارد هستید و نمی توانید به مستقیما روی کد های اصلی در master کار کنید. برای همین در این خط، باید یک شاخه *(branch)* جدید ایجاد کنید و در آن به توسعه بپردازید. اسم آن را `front` بگذارید.*

۴- حال لیست همه برنچ های موجود را مشاهده کنید.

۵- به برنچ `front` بروید

حال شروع به کار میکنید و تغییراتی را روی فایل index.html و myfile.js میدهید و این تغییرات را کامیت میکنید (نیازی به نوشتن این دستورات در پاسخ نیست).

۶و۷- به شاخه `master` بروید و شاخه `front` را با آن ادغام کنید.

۸- حال دیگر نیازی به شاخه `front` نداریم پس آنرا پاک کنید.

۹- تغییرات ایجاد کرده را در منبع ریموت آپلود کنید.

نکات تکمیلی:

- پیشنهاد میکنم حتما مراحل گفته شده در صورت سوال را امتحان و بررسی کنید و صرفا به نوشتن جواب ها کفایت نکنید.
- همه دستورات وارد شده باید مخصوص گیت باشند. پس همه دستورات با کلمه git شروع خواهند شد.
- شما برای پاسخ، باید فایلی به نام answer.txt ایجاد کرده و هر یک از دستورات را دقیقا به ترتیب نوشته شده در سوال بنویسید و ارسال کنید.
- پس فایل ارسالی شما **دقیقا دارای ۹ خط** خواهد بود.
- اگر جواب یک دستور را نمیدانید، آن خط را خالی بگذارید و به خط بعدی بروید.
- اگر همه نکات گفته شده در سوال را رعایت کردید و به جواب نهایی خود اطمینان دارید ولی نمره کامل را نگرفتید، میتوانید به من پیام دهید!