1. سلام و عرض ادب خدمت تمامی دوستانی که این فایل رو میخونند ، امیدوارم این فایل به شما کمک کنه که بتونید مواردی رو که میخواید رو یاد بگیرید یا حداقل ازش مثل یه دفترچه تلفن برای کاراتون استفاده کنید.  
   موردی که هست اینه که بنده در حال حاضر مهارتم در برنامه نویسی در حد جونیور هست پس   
   **اگر کم و کاست یا اشتباهی بود** به بزرگواری خودتون ببخشید  
   سعی کردم که به عامیانه ترین وساده ترین زبان ممکن این فایل رو بنویسم که حتی اگه خودمم چیزی رو فراموش کردم با یک مرور سریع بتونم مطالب رو بخاطر بیارم
2. گیت :   
   version controller
3. ریپازیتوری: محلی که تمام فایلات اونجان و توسط گیت قراره ردیابی بشن ( مثلا بتونه بهت هیستوری بده که چه تغییراتی دادی و...)
4. git init
5. اینجا میای صفر یک ریپوزیتوری رو مشخص میکنی
6. پیگیری تغییرات ایجاد شده :  
    git status
7. اضافه کردن فایل یا پوشه ای برای بررسی تغییرات :
8. git add <file/app name>
9. کامیت کردن : ایجاد کردن چک پوینت
10. git commit
11. اینجا برات از طریق ادیتوری که براش مشخص کردی یه فایل رو باز میکنه تو توی اولین خطش میای کامنت دلخواهتو مینویسی بعد همون فایل رو میبندی اینجوری چک پوینتت تثبیت میشه فقط قبلش باید git add رو زده باشی که اماده کامیت کردن شده باشه
12. این git add یجور حالت میانی هست که همونطور که گفتم فایل رو اماده میکنه برای کامیت کردن وگرنه تا وقتی که ادد نکنی نمیتونی کامیتش کنی
13. میتونی هم اینجوری بنویسی که دیگه کد ادیتورتو باز نکنی
14. git commit -m "your message will appear here"
15. اون دش -ام به معنای مسیج هست
16. دیدن تاریخچه تغییرات: با استفاده از دستور زیر میتونی تمام تغییراتی که از اول کار اعمال کردی تا الان رو ریز به ریز ببینی
17. git log
18. فایل gitignore. : میای یه فایل با همین اسم میسازی بعد هر ادرسی که توش قرار بدی (فایل یا پوشه) دیگه وقتی که git status میزنی که ببینی چیا رو کامیت نکردیو این حرفا دیگه نشونشون نمیده
19. code .gitignore
20. فقط وقتی که یه پوشه رو گذاشتی توش بایدبعد اسمش یه / قرار بدی که فکر نکنه فایله
21. اگه موقع کامیت کردن فراموش کردی و یه فایلو جا انداختی : میای اون فایل رو دوباره ادد میکنی بعدش از دستور
22. git commit —amend
23. استفاده میکنی ، فایل جدیدی که ادد کردی وارد کامیت قبلی میشه و دیگه نیازی نیس که کامیت جدید بسازی
24. کلون: میای یه ریپازیتوری (پروژه) رو دریافت میکنی و دیگه از اون نسخه ای که انتخاب میکنی به بعد شرو میکنی تغییرات رو خودت ایجاد میکنی
25. git clone <github url>
26. ریموت: لینک کردن ریپازیتوری محلی (تو کامپیوترت) با ریپازیتوری گیت هاب
27. git remote add <name> <github repo link>
28. برنچ (branch): یه انشعاب از روند خطی ایجاد شده ی اولیه برای اینکه چیزای دیگه رو هم امتحان کنیم
29. برای ساختن برنچ جدید از دستور زیر استفاده میکنیم
30. git <branch name>
31. برای اینکه از برنچی به برنچ دیگه تغییر محل بدیم از دستور زیر استفاده میکنیم
32. git switch <branch name>
33. برای اینکه ببینیم چه برنچ هایی داریم از دستور زیر استفاده میکنیم
34. git branch
35. برای تغییر دادن نام برنچ فعلی ای که توش هستیم از دستور زیر استفاده میکنیم
36. git branch -M <choose your name>
37. پوش (push): فرستادن اطلاعات لوکال به سرور
38. git push <remote name> <branch name>
39. گاهی اوقات ممکنه ما یه سری کامیت هایی انجام میدیم که هنوز روی سرور ثبت نشده برای اینکه بتونیم سرورمون رو هم اپدیت کنیم از دستور زیر استفاده میکنیم ( سرور میاد از متغیر (remote name/branch) تشخیص میده که چقدر کامیت های لوکالمون عقب تر هست)
40. git push <remote name> <branch name>
41. مرج (merge): انتقال اطلاعات برنچ دیگری به برنچ فعلی ای که در اون قرار داریم
42. git merge <branch name>
43. تو حالت عادی اگه برنچ مسترت کامیتی نداشته باشه به راحتی همون کامیتای برنچی که میخوای بیاریش تو مستر رو میاره و والد کامیت ایجاد شده همون اخرین کامیت برنچ مورد نظرت بود
44. اما اگه هم برنچی که میخوای بیاری کامیتای زیادی داشته (بیشتر از یکی) هم برنچ مسترت اون موقع مجبور میشه یه کامیت بسازه که دو تا والد داشته باشه (اخرین کامیت برنچ اصلیت و اون برنچی که ازش میخوای اطلاعاتو بیاری) که البته این مسیر خودکار انجام میشه نیازی نیست دستور خاصی رو اجرا کنی
45. موردی که ممکنه رخ بده اینه که تو برنچ های مختلف یک فایل با مشخصات یکسان اما محتوای یکسان باشه ، در اینجا هنگام مرج کردن بهت یه ارور نشون میده و فایل رو هم بهت میده (ادغام میکنه اطلاعات دو تا فایل رو ) بعد تو میری تو اون فایل و به صورت دستی اطلاعاتش رو درست میکنی که بدونی چی رو میخوای نگه داری چی رو میخوای حذف کنی بعدش دستور git commit رو اجرا میکنی و مرجت به صورت کامل انجام میشه
46. فتچ کردن (fetch): زمانی که سرور گیت هاب تغییراتش از تغییرات سیستم فعلی خودت جلو تره و خودت هم از طرفی یک یا چند تا کامیت انجام دادی که اطلاعات فعلی تو با اطلاعات فعلی سرور گیت هاب فرق میکنه ، در اینجا برای اینکه بتونی اطلاعات رو از سرور گیت هاب بیاره بدون اینکه به اطلاعات فعلیت دست بزنه باید از دستور زیر استفاده کنی ( یجورایی یه برنچ مجازی ایجاد میکنه و بهتم اطلاع میده که مثلا فلان تعداد کامیت تو عقب تری وقتی که از status استفاده کنی)
47. git fetch <remote> <branch>
48. پول کردن (pull): ترکیب فتچ کردن و مرج کردنه ، میره اطلاعات رو از سرور میگیره و بعد میاد با اطلاعات فعلی دستگاه خودت مرج میکنه
49. git pull <remote> <branch>