

11. โปรแกรม **Version control** มีประโยชน์อย่างไร
- ช่วยเราจัดเก็บการเปลี่ยนแปลงของโปรแกรมส่วนต่างๆเป็นเวอร์ชัน โดยเราสามารถเรียกมาดูเวอร์ชันเก่าๆได้ในภายหลัง
12. ข้อได้เปรียบของ **distributed version control** เมื่อเทียบกับ **centralized version control** คืออะไร
- ผู้ควบคุมสามารถควบคุมได้อย่างละเอียดว่าใครสามารถแก้ไขส่วนไหนได้บ้าง
13. ข้อได้เปรียบของ **centralized version control** เมื่อเทียบกับ **distributed version control** คืออะไร
- ผู้พัฒนาทุกคนได้รับ **Copy file** มาทั้งหมดของ **repo** และพัฒนาในเครื่องของตนเอง จึงทำให้ไม่มีความเสี่ยงที่จะทำงานไม่ได้เนื่องจากเซิร์ฟเวอร์ไม่สามารถใช้งานได้
14. บอกแนวทางในการแก้ไข **conflict** ที่เกิดขึ้นเมื่อมีการ **merge** โปรแกรมของผู้พัฒนาหลายๆคนเข้าด้วยกัน
- ใช้ **Branch** เข้ามาช่วยแก้ไขในการ **merge**
15. บอกแนวทางการลด **conflict** ที่จะเกิดขึ้นจากการ **merge** โปรแกรมของผู้พัฒนาหลายๆคนเข้าด้วยกัน
- ใช้ **Branch** เข้ามาช่วยแก้ไขในการ **merge**
16. **Git** คืออะไร แตกต่างจาก **Github** อย่างไร
- **Git** คือ **Version Control** ตัวหนึ่งมีหน้าที่ในการจัดเก็บการเปลี่ยนแปลงของไฟล์ ต่างจาก **Github** เพราะ **Github** เป็นแค่เซิร์ฟเวอร์ที่ไว้ให้บริการ **Git**
17. จุดประสงค์หลักในการ **branch** คืออะไร
- ลดปัญหาการ **Merge** , พัฒนา **Feature** ใหม่ๆที่ต่างจากตัว **master**
18. **Fast forward merge** คืออะไรและทำไมการ **push** ที่ **remote repo** จึงควรจะต้อง **merge** แบบนี้
- เป็นการ **merge** ที่รวม **branch** เข้าด้วยกัน จะ **merge** ก็ต่อเมื่อส่วนของ **branch** เสร็จเรียบร้อยแล้ว **branch** แล้ว
19. หน้าที่หลักของคำสั่ง **git pull** คืออะไร
- ดึงโปรแกรมจาก **repo** มารวมไว้ที่ **local**

20. แผนภาพด้านล่างนี้ต้องการสื่อความหมายอะไร

- เป็นแผนภาพอธิบายว่าในแต่ละ Version นั้น พัฒนาอะไรไปบ้าง เพิ่ม Feature อะไรบ้างเผยแพร่เมื่อไหร่