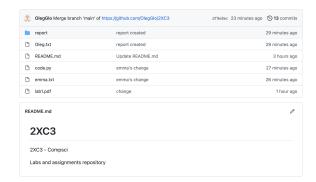
2XC3 - Lab 1 Report

Oleg Glotov L03, 400174037 glotovo@mcmaster.ca Emma Willson L02, 400309856 willsone@mcmaster.ca

January 22, 2022

1 Git Setup

The group began by setting up out github accounts and the repository. The visibility was set to private to comply with McMaster's code of conduct.



Both of us has access to the repository. We choose to work with github directly thorough our VSCode IDE since it fully integrates git commands and simplifies the workflow.

2 Git commands

Here we fucked around more resulting in the shit you see below

Here we have merge demo



Here we have reset demo

```
91 print(are_valid_groups(groups, studenthum))
92 print(are_valid_groups(groups, studenthum))
93 vnormalsize
94 vnodscoratora
95 vnormalsize
95 vsction(Player vs Adversary)
95 vsction(Player vs Adversary)
96 Let the games begin now
96 Let the games begin now reset demo
```

Here we have revert demo

```
usagie: git revert (=options=) <commit-ish>...

or: git revert <subcommand>

—quit
—continue resume revert or cherry-pick sequence
—abort cancel revert or cherry-pick sequence
—shopt skip current commit and continue
—cleanup <mode>
—n, —no-commit
—e, —edit don't automatically commit
—e, —edit edit the commit message
3 da Signed-off-by trailer
—n, —mainline <parent-numbers
select mainline parent
—rerere-autoupdate
—strategy <strategy
—X, —strategy-option
—spino for merge strategy
—5, —gpg-sign[=<key-ids]
—GG sign commit

olegglotov@Olegs-MacBook-Pro 2XC3 % git revert —quit
```

Write up revert and reset and

The

other

things

Merge demo

3 Code.py

Both group member's code is provided below. After consulting we decided to combine both of ours solution into the final product in the file code.py not provided here.

The merge interface was exactly similar to the one provided above but with more lines of code. We discussed both of our solutions and implemented both of our approaches into the final solution.

Oleg's version:

```
def are_valid_groups(groups, studentNum):
   length = len(studentNum)
   count = 0
   for group in groups:
       count = 0
       for students in studentNum:
            if (group.count(students) == 0):
                break
            if (group.count(students) == 1):
                count += 1
        if count == length:
            return True
   return False
print(are_valid_groups(groups, studentNum))
  Emma's version:
def are_valid_groups(nums, groups):
   valid = True
   for num in nums:
       valid = False
       for group in groups:
            if (num in group):
                valid = True
                break
        if (not valid):
            return False
   return True
print(are_valid_groups(groups, studentNum))
```

4 Player vs Adversary

Let the games begin now reset demo