[CS200]-Software Tools and Technologies Lab-II **16-09-2020**

Homework: 1

Team Name: Peaky Blinders Roll Numbers:11940150, 11940370, 11940380

Solution of problem 3. The script provided in this answer should be run after executing the script in problem 2 to get the correct output. The script is as follows:

```
cd Repo/q1                               # navigating to Repo/q1
git init                                 # initialize git
cd .git                                  # navigate to .git folder
cd objects                               # navigate to objects folder
for dir in *                             # go through all the git-objects
do
    cd $dir                              # navigate to the object folder
    for file in *                        # access the object file
    do
        str="$dir"                       # get first 2 digits of hash value
        if [ "$str" != "info" ]          # check if the folder is not info
        then
             if [ "$str" != "pack" ]     # check if the folder is not pack
             then
                str=$str"$file"          # get the full hash value
                git cat-file -t "$str"   # print type of the object
                git cat-file -p "$str"   # print content of the object
            fi
        fi
    done
    cd ..             # return to object folder to visit next object
done
```

The logic used is as follows: The SHA-1 function produces a 40 digit(hexadecimal) hash value for each object. The objects are stored in the folder .git/objects in the following manner: The sub-directory is named with the first 2 characters of the SHA-1, and the filename is the remaining 38 characters. Thus if we concatenate the folder and file name we get the hash value of the object. Using this hash value and the git cat-file function we can print the content(using -p) and type(using -t) of these objects as asked in the question.

The output(the types and the content of the git-objects) generated on my pc after running the script is given below:

```
tree
100644 blob 78201917ea4c7936828d74306e2c73181b18a92c    11940380

commit
tree 5ecbf368f467a189815a52926e47bbe8be57f318
parent acc31a73c5d1bd11c850c1ef9e572ea50f8821c7
```

```
author Dhruv <dhruva4000@gmail.com> 1600358670 +0530
committer Dhruv <dhruva4000@gmail.com> 1600358670 +0530

committed ritik

tree
040000 tree 1806845adbf93e33ea8d09425d8fada9715d9458    dhruv
040000 tree 67b7ada2039017b811fb95aea2ee83d05fee1755    ritik

tree
040000 tree 70f535e368bef1aa3f5d4c2297da0594e1e476ef    anubh
040000 tree 1806845adbf93e33ea8d09425d8fada9715d9458    dhruv
040000 tree 67b7ada2039017b811fb95aea2ee83d05fee1755    ritik

tree
100644 blob 78201917ea4c7936828d74306e2c73181b18a92c    11940370

tree
100644 blob 78201917ea4c7936828d74306e2c73181b18a92c    11940150

blob
iitbhilai

tree
040000 tree 1806845adbf93e33ea8d09425d8fada9715d9458    dhruv

commit
tree 847bafa663b1add2fce4e7c21b54bf420b4221bf
author Dhruv <dhruva4000@gmail.com> 1600358668 +0530
committer Dhruv <dhruva4000@gmail.com> 1600358668 +0530

committed dhruv

commit
tree 64d65a42e792c476e930b53c3709d463157ee85f
parent 57bd015ee9b506dd6bb43fc93655557cd50f1288
author Dhruv <dhruva4000@gmail.com> 1600358672 +0530
committer Dhruv <dhruva4000@gmail.com> 1600358672 +0530

committed anubh
```

As is seen from the list we have three commits, six trees and one blob. This is the exact contents of the final git-graph formed in question 2 after performing all the three commits. Thus we can conclude that the git-graph is basically a pictorial representation of the .git/objects folder.