These screenshots are for Task 1. From Task 1 to Task 3, it was pretty simple, until Task 4 where I had a lot of trouble.

Task 1

```
root@e1b8891446e3:/md5collgen# md5sum out1.bin out2.bin
673a4bc64ff7e48c0ba8159d98ef8a5a  out1.bin
673a4bc64ff7e48c0ba8159d98ef8a5a  out2.bin
root@e1b8891446e3:/md5collgen#
```

```
root@e1b8891446e3:/md5collgen# echo "This is the prefix content" > prefix.txt
root@e1b8891446e3:/md5collgen# ./md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 6625e60c73f79c9675d06e35e712ee52

Generating first block: .
Generating second block: S01.....................
Running time: 0.316448 s
root@e1b8891446e3:/md5collgen#
```

I wasn't too interested in the outcome of this task, the output was the same so I didn't care too much about it.

Task 2

```
root@e1b8891446e3:/md5collgen# echo "File 1 content" > file1
root@e1b8891446e3:/md5collgen# echo "File 2 content" > file2
root@e1b8891446e3:/md5collgen# echo "Suffix content" > suffix
root@e1b8891446e3:/md5collgen# cat file1 suffix > file1_with_suffix
root@e1b8891446e3:/md5collgen# cat file2 suffix > file2_with_suffix
root@e1b8891446e3:/md5collgen# md5sum file1_with_suffix file2_with_suffix
bccd3a8f76a8b31b4d5a8a7ef536590b  file1_with_suffix
4cd950babb7683d928b491f48f237337  file2_with_suffix
root@e1b8891446e3:/md5collgen#
```

This one was more interesting because the files were different. It felt like I actually did something different.

1. If the length of your prefix file is not multiple of 64, what is going to happen?

MD5 padding rules will apply. It automatically pads the prefix file with additional bytes to make its length a multiple of 64 bytes. The padded prefix will be used as input to the MD5 collision generation tool.**

2. Create a prefix file with exactly 64 bytes, and run the collision tool again, and see what happens. **There is no padding, P and Q are generated right, and the hashes are identical.**
3. Are the data (128 bytes) generated by md5collgen completely different for the two output files? Please identify all the bytes that are different.
   The differences are not spread across the whole 128. They are concentrated into specific areas because they are carefully made to cause a collision. A byte in block1 could be 0x7f and a byte in block2 could be 0x80.

Task 3

```
00000fe0: 0000 0000 0000 0000 0808 0000 0000 0000  ..........
00000ff0: ac07 0000 0000 0000 0000 0000 0000 0000  ..........
00001000: 0000 0000 0000 0000 0810 0100 0000 0000  ..........
00001010: 4242 4100 0000 0000 0000 0000 0000 0000  AAA.......
00001020: 0000 0000 0000 0000 0000 0000 0000 0000  ..........
00001030: 0000 0000 0000 0000 0000 0000 0000 0000  ..........
00001040: 0000 0000 0000 0000 0000 0000 0000 0000  ..........
00001050: 0000 0000 0000 0000 0000 0000 0000 0000  ..........
```

```
root@e1b8891446e3:/md5collgen# head -c 4112 program > prefix
root@e1b8891446e3:/md5collgen# dd if=program of=region bs=1 skip=4112 count=128
128+0 records in
128+0 records out
128 bytes copied, 0.000631833 s, 203 kB/s
root@e1b8891446e3:/md5collgen#
```

```
root@e1b8891446e3:/md5collgen# md5collgen -p prefix -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix'
Using initial value: d8e9ae27a8eee9ff42d04fdd4555a8f1

Generating first block: ......
Generating second block: S01..
Running time: 0.827104 s
root@e1b8891446e3:/md5collgen#
```

```
root@e1b8891446e3:/md5collgen# cat out1.bin region suffix > program1
root@e1b8891446e3:/md5collgen# cat out2.bin region suffix > program2
root@e1b8891446e3:/md5collgen# md5sum program1 program2
3649bf6363e93da38be366cc488cc57e  program1
3649bf6363e93da38be366cc488cc57e  program2
root@e1b8891446e3:/md5collgen#
```

```
root@e1b8891446e3:/md5collgen# xxd -r program.hex program_modified
root@e1b8891446e3:/md5collgen# chmod +x program_modified
root@e1b8891446e3:/md5collgen# ./program_modified
4242410000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0
root@e1b8891446e3:/md5collgen#
```

root@e1b8891446e3:/md5collgen# ./program1

00000000000000000000000000000000000000000000000dd1eb6267d3475c62482d699
855c32b5c4f371a128892531f327fe97340a4a3e142d1fc91edf3f8eb19c99dd96c32c664c6
fed5d74733c8745928a2985b992928d85796999d33050d944e7e6da5ef3291f461ee5d35a7
63bec46fee3569dc6d8f4ca38dca627837d9787a2e35e0e42d7255ccf94ae8246456d1c2c4
1414100000000000000000000

root@e1b8891446e3:/md5collgen# ./program2

00000000000000000000000000000000000000000000000dd1eb6267d3475c62482d699
855c32b5c4f3712128892531f327fe97340a4a3e142d1fc91edf3f8eb19c91dda6c32c664c6
fed5d74733c87c5928a2985b992928d85796999d33050d944e7e6da5ef3a91f461ee5d35a7
63bec46fee3569dc6d8f4ca38dca627837d9f8792e35e0e42d7255ccf94ae82c6456d1c2c4
1414100000000000000000000

This step was weird for me because I thought I was doing it wrong, until I saw the next step where I actually had to make the hashes the same and the behavior different.

Task 4

```
root@e1b8891446e3:/md5collgen# xxd program1 | less
root@e1b8891446e3:/md5collgen# xxd program1 > program1.hex
root@e1b8891446e3:/md5collgen# xxd program2 > program2.hex
root@e1b8891446e3:/md5collgen# md5sum program1 program2
3649bf6363e93da38be366cc488cc57e  program1
3649bf6363e93da38be366cc488cc57e  program2
root@e1b8891446e3:/md5collgen#
```

```
root@e1b8891446e3:/md5collgen# ./program1
Malicious behavior
root@e1b8891446e3:/md5collgen# ./program2
Malicious behavior
root@e1b8891446e3:/md5collgen#
```

I don't know what I did but I just couldn't get the programs to do benign behavior and malicious behavior. Only 1 or the other. Overall, this lab was pretty fun, I couldn't get the last step to work but it was interesting to see how the collisions worked with the files, the programs and the converting to hex and binary, and all the things I had to install into the environment. Unfortunately, there were a lot of different commands I had to do to fix issues and download a lot of things, but it was fun.