a)

$A(D, q)$ How A works: $q(D) + Lap(0, \varepsilon)$

$$\downarrow \quad \frac{1}{2\varepsilon} e^{\left(-\frac{|x-0|}{\varepsilon}\right)}$$

proving that algorithm A satisfies ε-DP is the same as proving that $Lap(0, \varepsilon)$ satisfies ε-DP.

$$\frac{Pr[A(D, q) = O]}{Pr[A(D', q) = O]} = \frac{Pr[q(D) + Lap(0, \varepsilon) = O]}{Pr[q(D') + Lap(0, \varepsilon) = O]}$$

$$= \frac{\cancel{\frac{1}{2\varepsilon}} e^{\left(-\frac{|O - q(D)|}{\varepsilon}\right)}}{\cancel{\frac{1}{2\varepsilon}} e^{\left(-\frac{|O - q(D')|}{\varepsilon}\right)}} = e^{\left(-\frac{|O - q(D)|}{\varepsilon} + \frac{|O - q(D')|}{\varepsilon}\right)}$$

$$= e^{\frac{|\cancel{O} - q(D') - \cancel{O} + q(D)|}{\varepsilon}} = e^{\frac{|q(D) - q(D')|}{\varepsilon}}$$

A satisfies DP whenever $|q(D) - q(D')|$, that is the sensitivity of the query, is less than or equal to ε.

b)

$$\frac{Pr[A(v_1)=y]}{Pr[A(v_2)=y]} = \frac{\dfrac{e^{\frac{-\alpha\cdot d(v_1,y)}{2}}}{\sum_{z\in U} e^{\frac{-\alpha\cdot d(v_1,z)}{2}}}}{\dfrac{e^{\frac{-\alpha\cdot d(v_2,y)}{2}}}{\sum_{z\in U} e^{\frac{-\alpha\cdot d(v_2,z)}{2}}}}$$

$$= \frac{e^{\frac{-\alpha\cdot d(v_1,y)}{2}}}{\sum_{z\in U} e^{\frac{-\alpha\cdot d(v_1,z)}{2}}} \cdot \frac{\sum_{z\in U} e^{\frac{-\alpha\cdot d(v_2,z)}{2}}}{e^{\frac{-\alpha\cdot d(v_2,y)}{2}}}$$

triangle inequality
$$\begin{cases} d(v_1,y) \le d(v_1,v_2) + d(v_2,y) \\ d(v_2,z) \le d(v_2,v_1) + d(v_1,z) \\ d(v_1,z) \ge d(v_2,z) - d(v_2,v_1) \end{cases}$$

$$\le \frac{e^{\frac{-\alpha\cdot(d(v_1,v_2)+d(v_2,y))}{2}}}{\sum_{z\in U} e^{\frac{-\alpha\cdot(d(v_2,z)-d(v_1,v_2))}{2}}} \cdot \frac{\sum_{z\in U} e^{\frac{-\alpha\cdot d(v_2,z)}{2}}}{e^{\frac{-\alpha\cdot d(v_2,y)}{2}}}$$

$$= \frac{e^{\frac{-\alpha d(v_1,v_2)}{2}} \cdot \cancel{e^{\frac{-\alpha d(v_2,y)}{2}}}}{\sum_{z\in U} e^{\frac{-\alpha d(v_2,z)}{2}} \cdot e^{\frac{\alpha d(v_1,v_2)}{2}}} \cdot \frac{\sum_{z\in U} e^{\frac{-\alpha\cdot d(v_2,z)}{2}}}{\cancel{e^{\frac{-\alpha\cdot d(v_2,y)}{2}}}}$$

↳ take out, constant

$$= \frac{e^{\frac{-\alpha\, d(v_1, v_2)}{2}}}{e^{\frac{\alpha\, d(v_1, v_2)}{2}} \cdot \cancel{\sum_{z \in U} e^{\frac{-\alpha\, d(v_2, z)}{2}}}} \cdot \cancel{\sum_{z \in U} e^{\frac{-\alpha \cdot d(v_2, z)}{2}}}$$

$$= \frac{e^{\frac{-\alpha\, d(v_1, v_2)}{2}}}{e^{\frac{\alpha\, d(v_1, v_2)}{2}}} = e^{\frac{-\alpha\, d(v_1, v_2)}{2} - \frac{\alpha\, d(v_1, v_2)}{2}}$$

$$= e^{\frac{-\alpha\, d(v_1, v_2) - \alpha\, d(v_1, v_2)}{2}}$$

$$= e^{\frac{-\cancel{2}\alpha\, d(v_1, v_2)}{\cancel{2}}} = e^{-\alpha\, d(v_1, v_2)}$$

$$e^{-\alpha\, d(v_1, v_2)} \leq e^{\alpha\, d(v_1, v_2)}$$

Part 2.

a)

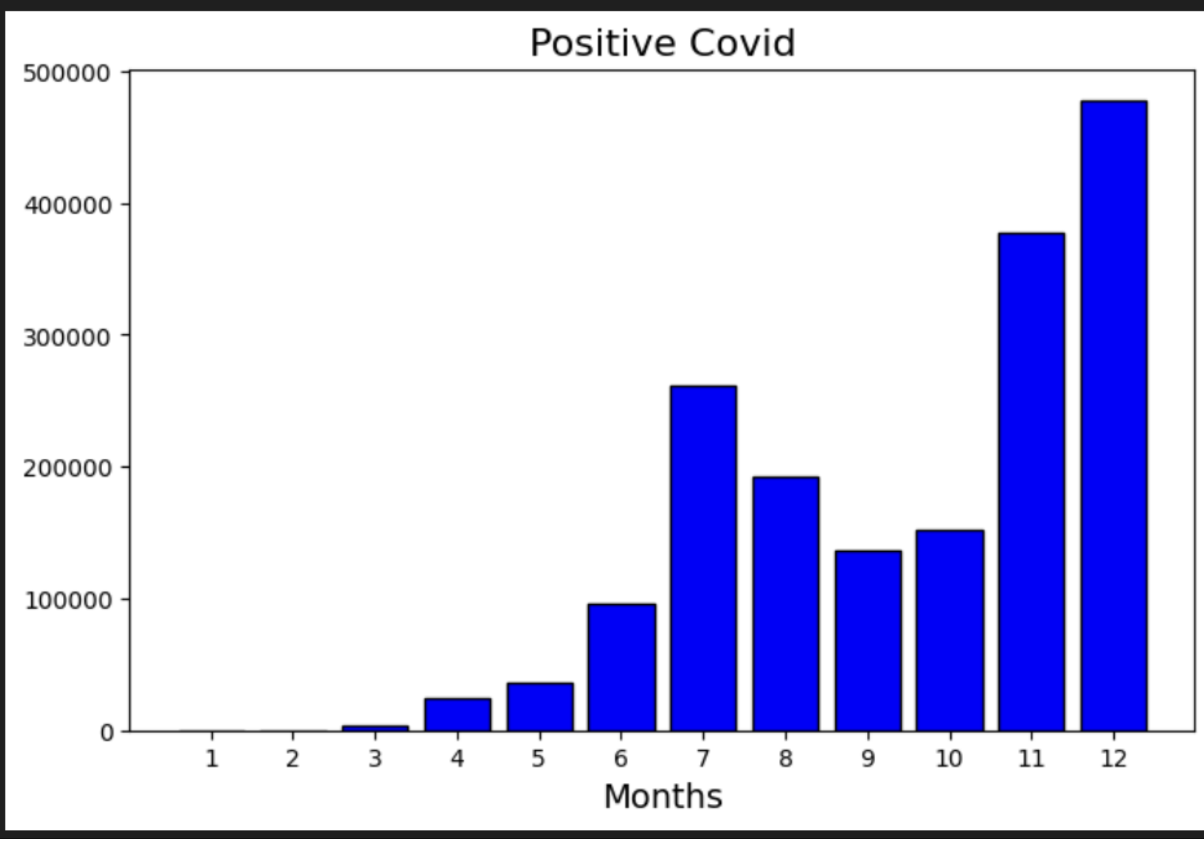
```python
import matplotlib.pyplot as plt
heights = get_histogram('./covid19-states-history.csv')

x_labels = range(1, 13)

plt.figure(figsize=(8, 5))
plt.bar(x_labels, heights, color='blue', edgecolor='black')

plt.title("Positive Covid", fontsize=16)
plt.xlabel("Months", fontsize=14)
plt.xticks(x_labels)
plt.show()
```

✓ 0.1s



d)

```python
    print("**** LAPLACE EXPERIMENT RESULTS ****")
    eps_values = [0.0001, 0.001, 0.005, 0.01, 0.05, 0.1, 1.0]
    error_avg = epsilon_experiment(dataset, state, year, eps_values, 2)
    for i in range(len(eps_values)):
        print("eps = ", eps_values[i], " error = ", error_avg[i])

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
**** LAPLACE EXPERIMENT RESULTS ****
eps =  0.0001  error =  22001.377777848706
eps =  0.001  error =  1927.876558052895
eps =  0.005  error =  459.4158128532987
eps =  0.01  error =  190.3625565993274
eps =  0.05  error =  37.407812075674954
eps =  0.1  error =  20.5438005934323
eps =  1.0  error =  2.0651323591776864
```

As we can see, the higher is the epsilon value – the lower is the difference between our estimates and the actual data. That is because the higher epsilon values mean higher loss of privacy (more budget).

e)

```python
    print("**** N EXPERIMENT RESULTS ****")
    N_values = [1, 2, 4, 8]
    error_avg = N_experiment(dataset, state, year, 0.5, N_values)
    for i in range(len(N_values)):
        print("N = ", N_values[i], " error = ", error_avg[i])
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```
**** N EXPERIMENT RESULTS ****
N =  1  error =  1.8189558989300605
N =  1  error =  1.8189558989300605
N =  2  error =  3.441642829649537
N =  4  error =  9.456151744768569
N =  8  error =  15.838642402982066
```

The higher is N the more often an individual can test positive for covid – that is the higher variation in data there can be so the error increases.

g)

```python
print("**** EXPONENTIAL EXPERIMENT RESULTS ****")
eps_values = [0.0001, 0.001, 0.01, 0.05, 0.1, 1.0]
exponential_experiment_result = exponential_experiment(dataset, state, year, eps_values)
for i in range(len(eps_values)):
    print("eps = ", eps_values[i], " accuracy = ", exponential_experiment_result[i])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
**** EXPONENTIAL EXPERIMENT RESULTS ****
eps =  0.0001  accuracy =  6.8
eps =  0.001  accuracy =  9.2
eps =  0.01  accuracy =  21.7
eps =  0.05  accuracy =  88.0
eps =  0.1  accuracy =  99.1
eps =  1.0  accuracy =  100.0
```

The higher is epsilon the more data is given up so the higher is the accuracy, therefore higher epsilon leads to higher accuracy.

Part 3.

```
GRR EXPERIMENT
e=0.1, Error: 13861.07
e=0.5, Error: 3738.56
e=1.0, Error: 1623.06
e=2.0, Error: 463.40
e=4.0, Error: 106.20
e=6.0, Error: 40.21
RAPPOR EXPERIMENT
e=0.1, Error: 9537.90
e=0.5, Error: 2095.35
e=1.0, Error: 1059.85
e=2.0, Error: 508.81
e=4.0, Error: 244.83
e=6.0, Error: 97.72
OUE EXPERIMENT
e=0.1, Error: 10999.29
e=0.5, Error: 2375.94
e=1.0, Error: 1332.93
e=2.0, Error: 561.32
e=4.0, Error: 194.61
e=6.0, Error: 124.62
```

As we can see there is no protocol that is better than the rest. The accuracy of estimates increases as epsilon increases because privacy decreases -> resulting in more information being given up to the mechanism.