P1.

```python
#!/usr/bin/env python3
""" LPC stats HW2, Problem 1
    Author: Caleb Fangmeier
    Created: Sep. 25, 2017
"""
from numpy import sum, sqrt
from numpy.random import exponential, poisson


def find_relative_frequency(n_experiments, b):
    # First get the mean counts for each experiment
    a = exponential(scale=b, size=n_experiments)

    # Since this is a counting experiment, sample from a Poisson distribution with the
    # previously generated means.
    x = poisson(lam=a)

    # Finally, count for how many experiments the a lies in
    # the range [x-sqrt(x), x+sqrt(x)]
    n_pass = sum((x - sqrt(x) < a) & (a < x + sqrt(x)))
    relative_frequency = n_pass / n_experiments

    # and print the results
    print(f"For b={b}, {n_pass}/{n_experiments} passed, R={relative_frequency*100:4.2f}%")


find_relative_frequency(10000, 5)
find_relative_frequency(10000, 10)

# Example output
# > For b=5, 6012/10000 passed, R=60.12%
# > For b=10, 6372/10000 passed, R=63.72%
```

P2.

a) We start with

$$p(D|s) = \int_0^\infty p(D|s,b)\,\pi(b|s)\,db \qquad (1)$$

for π, we know the background count has mean Kb and observed count M so,

$$\pi(b|s) = \text{poisson}(Kb, M).$$

for $p(D|s,b)$, we know the total count has mean $s+b$ and observed coount N so,

$$p(D|s,b) = \text{poisson}(s+b, N).$$

combinig these into (1) gives

$$p(D|s) = \int_0^\infty \left(\frac{(s+b)^N}{N!} e^{-(s+b)}\right)\left(\frac{(Kb)^M e^{-Kb}}{M!}\right) db$$

$$= \frac{1}{N!\,M!} e^{-s} \int_0^\infty e^{-(1+K)b} (s+b)^N (Kb)^M \, db$$

Now we can use the binomial theorem to expand $(s+b)^N$ to $\sum_{r=0}^{N} \binom{N}{r} s^{N-r} b^r$, and with a bit of rearrengement we get.

$$p(D|s) = \sum_{r=0}^{N} \left(\frac{s^{N-r} e^{-s}}{(N-r)!}\right) \frac{(1-x)^M}{x^M\, r!\, M!} \int_0^\infty b^{r+M} e^{-b/x} \, db$$

Next, we can use an integral table to find

$$\int_0^\infty b^{r+M} e^{-b/x} = (r+M)!\, x^{M+r+1}.$$

which, again, after a bit of manipulation yields

$$p(D|s) = \frac{x(1-x)}{M} \sum_{r=0}^{N} \left(\frac{s^{N-r} e^{-s}}{(N-r)!}\right)\left(\frac{(M+r)!}{r!\,(M-1)!} x^{r+1} (1-x)^M\right)$$

Finally, we recognize the first term in parentheses as the poisson pdf and the second as the beta pdf.

$$p(D|s) = \frac{x(1-x)}{M} \sum_{r=0}^{N} \text{poisson}(N-r, s)\ \text{beta}(x, r+1, M)$$

P2b.

```python
#!/usr/bin/env python3
""" LPC stats HW2, Problem 2
    Author: Caleb Fangmeier
    Created: Sep. 27, 2017
"""
from scipy.stats import beta, poisson
from scipy.optimize import root

# Declare the experimental results
N = 25
M = 353
k = 37.6
x = 1/(1+k)

p = 0.683


def DL(z):
    """
        This is just the DL function as defined in the problem statement. It is the
        cumulative distribution function for p(D|s).
    """
    num = 0
    den = 0
    for r in range(N+1):
        b = beta.pdf(x, r+1, M)
        num += b*poisson.cdf(N-r+1, z)
        den += b
    return num/den


def DR(z):
    """
       Same as DL, but for the opposite side.
    """
    return 1 - DL(z)


# Scipy's root function finds a zero crossing for the given function.
# The second argument is just the starting point for the algorithm.
root_DL = root(lambda z: DL(z) - (1-p)/2, 10).x[0]
root_DR = root(lambda z: DR(z) - (1-p)/2, 10).x[0]
print(f"The interval is: [{root_DR:.2f}, {root_DL:.2f}]")

# Output
# > The interval is: [12.39, 22.78]
```