5.8 PROJECTS

Project 5.1 The magic of polling

> According to the latest poll, the president's job approval rating is at 45%, with a margin of error of ±3%, based on interviews with approximately 1,500 adults over the weekend.

We see news headlines like this all the time. But how can a poll of 1,500 randomly chosen people claim to represent the opinions of millions in the general population? How can the pollsters be so certain of the margin of error? In this project, we will investigate how well random sampling can really estimate the characteristics of a larger population. We will assume that we know the true percentage of the overall population with some characteristic or belief, and then investigate how accurate a much smaller poll is likely to get.

Suppose we know that 30% of the national population agrees with the statement, "Animals should be afforded the same rights as human beings." Intuitively, this means that, if we randomly sample ten individuals from this population, we should, on average, find that three of them agree with the statement and seven do not. But does it follow that every poll of ten randomly chosen people will mirror the percentage of the larger population? Unlike a Monte Carlo simulation, a poll is taken just once (or maybe twice) at any particular point in time. To have confidence in the poll results, we need some assurance that the results would not be drastically different if the poll had queried a different group of randomly chosen individuals. For example, suppose you polled ten people and found that two agreed with the statement, then polled ten more people and found that seven agreed, and then polled ten more people and found that all ten agreed. What would you conclude? There is too much variation for this poll to be credible. But what if we polled more than ten people? Does the variation, and hence the trustworthiness, improve?

In this project, you will write a program to investigate questions such as these and determine empirically how large a sample needs to be to reliably represent the sentiments of a large population.

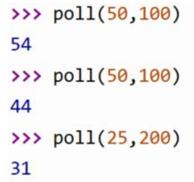
1. Simulate a poll

In conducting this poll, the pollster asks each randomly selected individual whether he or she agrees with the statement. We know that 30% of the population does, so there is a 30% chance that each individual answers "yes." To simulate this polling process, we can iterate over the number of individuals being polled and count them as a "yes" with probability 0.3. The final count at the end of the loop, divided by the number of polled individuals, gives us the poll result. Implement this simulation by writing a function

```
poll(percentage, pollSize)
```

that simulates the polling of `pollSize` individuals from a large population in which the given `percentage` (between 0 and 100) will respond "yes." The function should return the percentage (between 0 and 100) of the poll that actually responded "yes."

Remember that the result will be different every time the function is called. Test your function with a variety of poll sizes.



```
>>> poll(50,100)
54
>>> poll(50,100)
44
>>> poll(25,200)
31
```

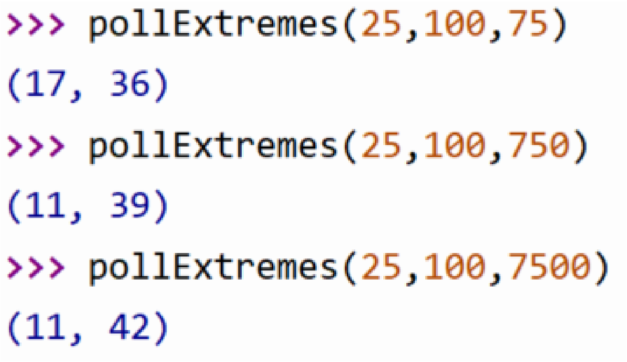
2. Find the polling extremes

To investigate how much variation there can be in a poll of a particular size, write a function

```
pollExtremes(percentage, pollSize, trials)
```

that builds a list of `trials` poll results by calling `poll(percentage, pollSize)` `trials` times. The function should return the minimum and maximum percentages in this list. For example, if five trials give the percentages `[28, 35, 31, 24, 31]`, the function should return the minimum `24` and maximum `35`.

Test your function with a variety of poll sizes and numbers of trials.



```
>>> pollExtremes(25,100,75)
(17, 36)
>>> pollExtremes(25,100,750)
(11, 39)
>>> pollExtremes(25,100,7500)
(11, 42)
```

3. What is a sufficient poll size?

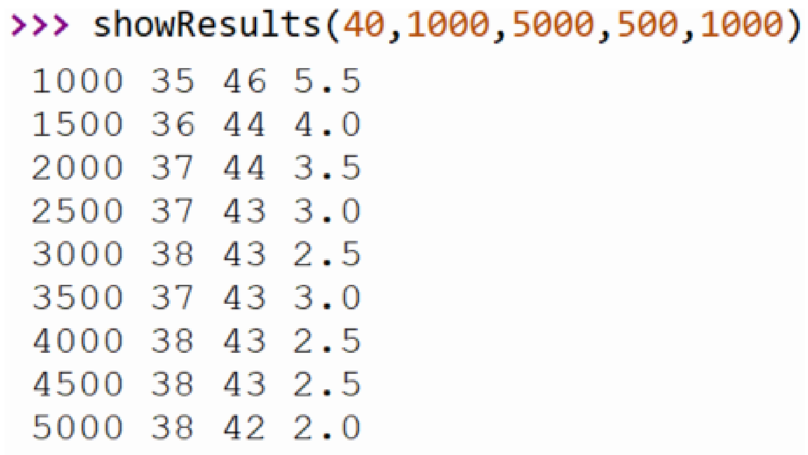
Next, we want to use your previous functions to investigate how increasing poll sizes affect the variation of the poll results. Intuitively, the more people you poll, the more accurate the results should be. Write a function

```
showResults(percentage, minPollSize, maxPollSize, step, trials)
```

that prints the minimum and maximum percentages returned by calling the function `pollExtremes(percentage, pollSize, trials)` for values of `pollSize` ranging from `minPollSize` to `maxPollSize`, in increments of `step`. For each poll size, call your `pollExtremes` function with

```
low, high = pollExtremes(percentage, pollSize, trials)
```

and then print the values of `low` and `high` along with the margin of error, defined to be `(high -low)` / 2.



```
>>> showResults(40,1000,5000,500,1000)
 1000 35 46 5.5
 1500 36 44 4.0
 2000 37 44 3.5
 2500 37 43 3.0
 3000 38 43 2.5
 3500 37 43 3.0
 4000 38 43 2.5
 4500 38 43 2.5
 5000 38 42 2.0
```