1.

(a) It is not (or should not be) a surprise. By 95% confidence, what we are claiming is that our process has a 95 percent success rate of providing boundaries that enclose the population parameter—in this case, $\mu = 70$, the center of the population.

(b) The 95% z^* critical value is found using the command

```
qnorm(0.975)

[1] 1.959964
```

(c)
```
enn <- 7
zStar <- qnorm(0.975)
runResults <- do(10000) * {
    sampledVals <- rnorm(enn, 70, 3)
    xbar <- mean(~sampledVals)
    ess <- sd(~sampledVals)
    myCI <- xbar + c(-1, 1) * zStar * ess/sqrt(enn)
    isInside(70, myCI)
}
prop(runResults == TRUE)  # gives the effective coverage rate of the simulation

prop_TRUE
    0.902
```

Because we obtained our critical value from the wrong distribution (normal instead of Student-t), the width of the interval is wrong. It is, in fact, less wide, as the standard normal distribution has less area in its tails than t-distributions. This is felt in seeing an effective coverage rate that isn't as large as "advertised".

(d)
```
enn <- 7
tStar <- qt(0.975, df = enn - 1)
runResults <- do(10000) * {
    sampledVals <- rexp(enn, 0.02)
    xbar <- mean(~sampledVals)
    ess <- sd(~sampledVals)
    myCI <- xbar + c(-1, 1) * zStar * ess/sqrt(enn)
    isInside(50, myCI)
}
prop(runResults == TRUE)  # gives the effective coverage rate of the simulation

prop_TRUE
   0.8432
```

The coverage rate is lower than desired (95% was the target), and this is due to the combination of skewed population and small sample size.

(e)
```
enn <- 35
tStar <- qt(0.975, df = enn - 1)
runResults <- do(10000) * {
    sampledVals <- rexp(enn, 0.02)
    xbar <- mean(~sampledVals)
    ess <- sd(~sampledVals)
    myCI <- xbar + c(-1, 1) * tStar * ess/sqrt(enn)
    isInside(50, myCI)
}
prop(runResults == TRUE)  # gives the effective coverage rate of the simulation

prop_TRUE
   0.9303
```

The coverage rate is not quite 95%, as was sought, but it is back up close to that amount, and this despite that we are still drawing from a skewed population. Nevertheless, the sample size $n = 35$ is more than 30, so the rule of thumb has us much closer to the target rate.

(f)
```
enn <- 7
tStar <- qt(0.95, enn - 1)
runResults <- do(10000) * {
    sampledVals <- rnorm(enn, 70, 3)
    xbar <- mean(~sampledVals)
    ess <- sd(~sampledVals)
    myCI <- xbar + c(-1, 1) * zStar * ess/sqrt(enn)
    isInside(70, myCI)
}
prop(runResults == TRUE)  # gives the effective coverage rate of the simulation

prop_TRUE
   0.9028
```

This effective coverage rate is quite close to the target rate of 90%.

2.

(a) Consider the quantitative variable from `NHANES` called `AgeMonths`. I use `diffmean()` to obtain the observed difference in sample means $\bar{x}_Y - \bar{x}_N$, which serves as a point estimate for $\mu_Y - \mu_N$:

```
diffmean(AgeMonths ~ SleepTrouble, data = NHANES, na.rm = TRUE)

diffmean
66.72255
```

It is OK to insert a different variable in place of `AgeMonths`. we can obtain a single bootstrap statistic by resampling separately from those respondents who have sleep trouble and those who do not:

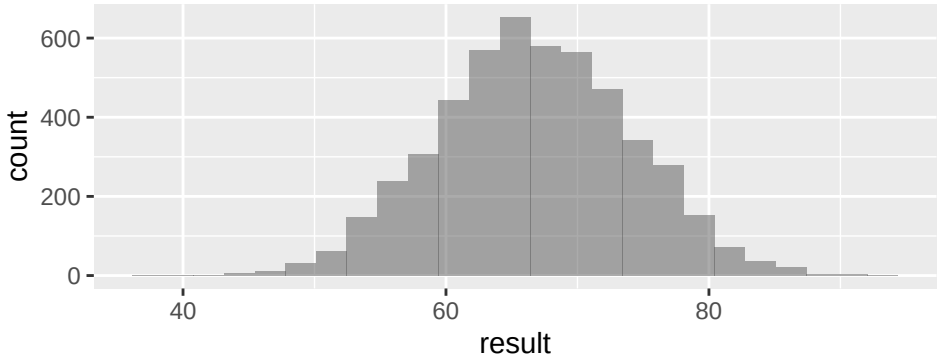
```
subset(NHANES, SleepTrouble == "Yes") -> sleepTroubleYes
subset(NHANES, SleepTrouble == "No") -> sleepTroubleNo
mean(~AgeMonths, data = resample(sleepTroubleYes), na.rm = TRUE) - mean(~AgeMonths, data = resample
    na.rm = TRUE)

[1] 72.83317

manyDiffs <- do(5000) * (mean(~AgeMonths, data = resample(sleepTroubleYes), na.rm = TRUE) -
    mean(~AgeMonths, data = resample(sleepTroubleNo), na.rm = TRUE))
```

We can view this bootstrap distribution:

```
gf_histogram(~result, data = manyDiffs)
```



But, its real use in finding a 95% CI is

- in offering us an means for approximating the standard error

  ```
  stdErrDiffOfMeans <- sd(~result, data = manyDiffs)
  ```

  which we can use, along with our point estimate and a critica value, to construct the confidence interval

  ```
  66.723 + c(-1, 1) * qt(0.975, df = 1972) * stdErrDiffOfMeans

  [1] 52.36893 81.07707
  ```

  Here, I have used the conservative formula for df.

- in offering us the ability to find the 0.025 and 0.975-quantiles:

  ```
  qdata(~result, data = manyDiffs, p = c(0.025, 0.975))

      2.5%    97.5%
  52.68689 80.82681
  ```

Either of these is an acceptable answer as a 95% bootstrap CI.

(b) Let n_Y represent the number out of the total who say "yes" to having sleep trouble, and n_N the subjects who say "no".

To generate one bootstrap statistic $\bar{x}_Y - \bar{x}_N$, we write the `AgeMonths` values for all the "yes" sleep-troubled subjects on slips and place them in a bag. That bag now contains n_Y (917 of them) slips. We would sample with replacement from that bag exactly n_Y times, record the numbers, and compute

a mean $\bar{x}_Y$. Similarly, in another bag, we would place slips with the `AgeMonths` values for the "no" sleep-troubled subjects, drawn with replacment from that bag $n_N = 2856$ times, and compute that mean $\bar{x}_N$. The bootstrap statistic is the difference of these, $\bar{x}_Y - \bar{x}_N$.

(c) The command

```
t.test(AgeMonths ~ SleepTrouble, data = NHANES)
```

produces a 95% confidence interval. We cannot easily control which way it choose to subtract the means (i.e, whether it takes $\bar{x}_Y - \bar{x}_N$ or $\bar{x}_N - \bar{x}_Y$ as its point estimate), and it automatically uses the Welch formula for df. So as to be able to compare with the approache in part (a) under the same df, I construct this CI using distinct steps:

```
ptEst <- diffmean(AgeMonths ~ SleepTrouble, data = NHANES, na.rm = TRUE)
s1 <- sd(~AgeMonths, data = sleepTroubleYes, na.rm = TRUE)
n1 <- nrow(sleepTroubleYes)
s2 <- sd(~AgeMonths, data = sleepTroubleNo, na.rm = TRUE)
n2 <- nrow(sleepTroubleNo)
stdError <- sqrt(s1^2/n1 + s2^2/n2)
tstar <- qt(0.975, df = 1972)
ptEst + c(-1, 1) * tstar * stdError

[1] 56.87813 76.56697
```

3.

(a) I consider, again, the quantitative variable `AgeMonths`. Hypotheses we test:

$$\mathbf{H}_0 : \ \mu_Y - \mu_N = 0, \qquad \mathbf{H}_a : \ \mu_Y - \mu_N \neq 0$$

We obtain a randomization distribution by scrambling the connection between `AgeMonths` and `SleepTrouble` values, given that the null hypothesis is another guise for asserting *no association* between these variables:

```
manyRandDiffs <- do(5000) * diffmean(AgeMonths ~ shuffle(SleepTrouble), data = NHANES, na.rm = TRUE
```

Our test statistic is

```
diffmean(AgeMonths ~ SleepTrouble, data = NHANES, na.rm = TRUE)

diffmean
66.72255
```

The P-value corresponds to the relative frequency of values as extreme or even more so than our test statistic:

```
nrow(filter(manyRandDiffs, abs(diffmean) >= 66.7225))/5000

[1] 0
```

(b) Imagine putting the `AgeMonths` value of every subject on a slip of paper and mixing them well in a bag. If we began drawing slips out, without replacement, we could take the first n_Y drawn slips (in this case, 917 of them) as representing the "yes" sleep-trouble group, and calculate the mean `AgeMonths` value for those slips, calling it $\overline{x}_Y$. There would be n_N slips (2856 of them) remaining, that would make the "no" sleep-trouble group, and have mean `AgeMonths` value $\overline{x}_N$. Having computed these, we subtract the to get one randomization statistic $\overline{x}_Y - \overline{x}_N$.

(c) The approach that employs formulas requires us to standardize the test statistic, then employ an appropriate t-distribution to evaluate the P-value. Here, I again use the conservative formula to obtain $df = 1973 - 1 = 1972$.

```
standardizedTestStat = 66.72255/stdError
2 * (1 - pt(standardizedTestStat, df = 1972))

[1] 0
```