Lab: Hypothesis Testing

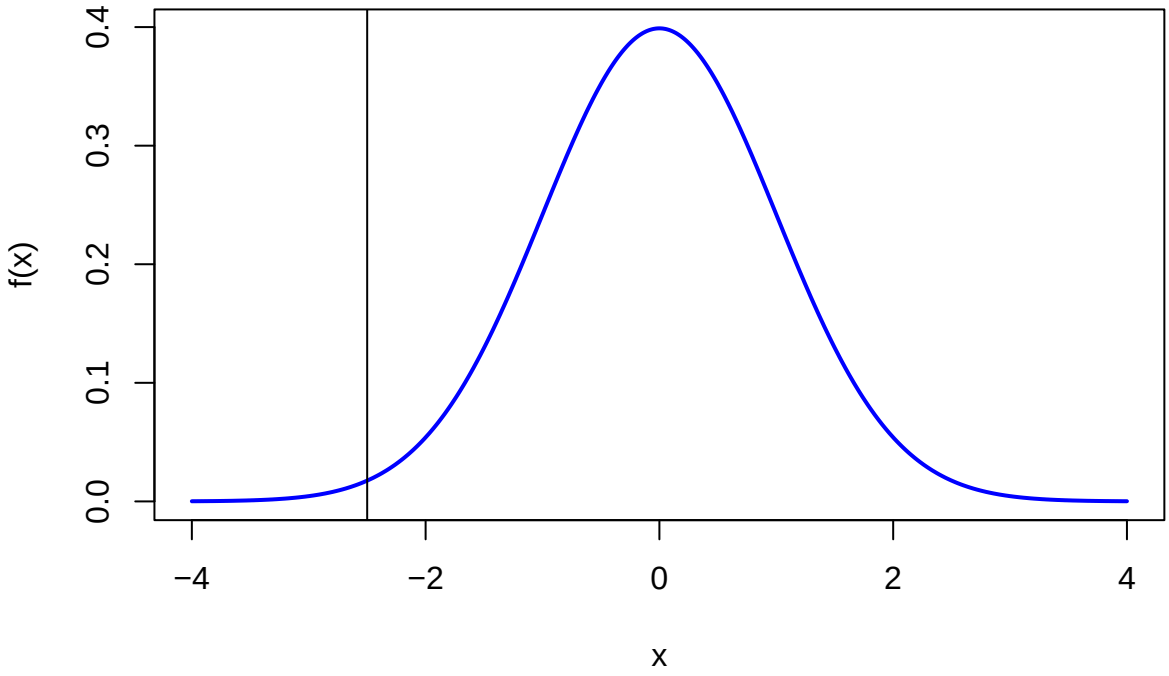
36-600

Fall 2022

Question 1

Let's assume for a moment that you are performing a lower-tail large-sample z test where the null hypothesis is $\mu_o = 6$, the sample mean is $\bar{X} = 5$, the sample standard deviation is $S = 4$, and the sample size is $n = 100$. Also, assume that you will tolerate a Type I error $\alpha = 0.05$. In the extended example in the notes, we did not standardize, but do that here: plot a standard normal distribution, and add vertical lines where the observed test statistic lies and where the boundary of the rejection region lies, i.e., the line to the left of which the area under the curve is α. (The notes provide an example of plotting and then adding vertical lines... make use of that example! The test statistic is on slide 10.)

```
n <- 100
z <- (5-6) / (4/sqrt(n))
fx = dnorm(seq(-4,4,by=0.01))
plot(seq(-4,4,by=0.01), fx, typ="l", col="blue", lwd=2, xlab="x", ylab="f(x)")
abline(v=z)
```



Question 2

For the set-up in Question 1, what is the p-value? What is your conclusion?

```
p <- pnorm(z)
p
```

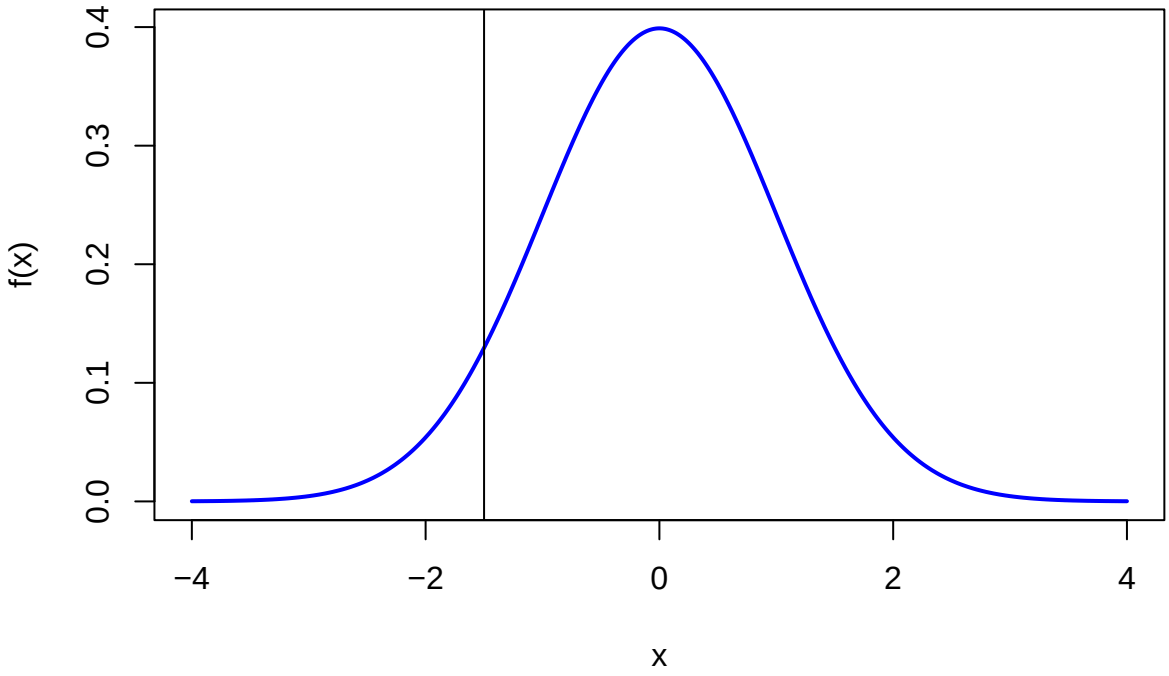
```
## [1] 0.006209665
```

The p-value is 0.006209665, which is less than 0.05, so we reject the null hypothesis that $\mu_o = 6$.

Question 3

Assume that $n = 36$, i.e., you have a smaller sample size, with all other quantities being the same. What is your conclusion now?

```
n <- 36
z <- (5-6) / (4/sqrt(n))
fx = dnorm(seq(-4,4,by=0.01))
plot(seq(-4,4,by=0.01), fx, typ="l", col="blue", lwd=2, xlab="x", ylab="f(x)")
abline(v=z)
```




```
p <- pnorm(z)
p
```

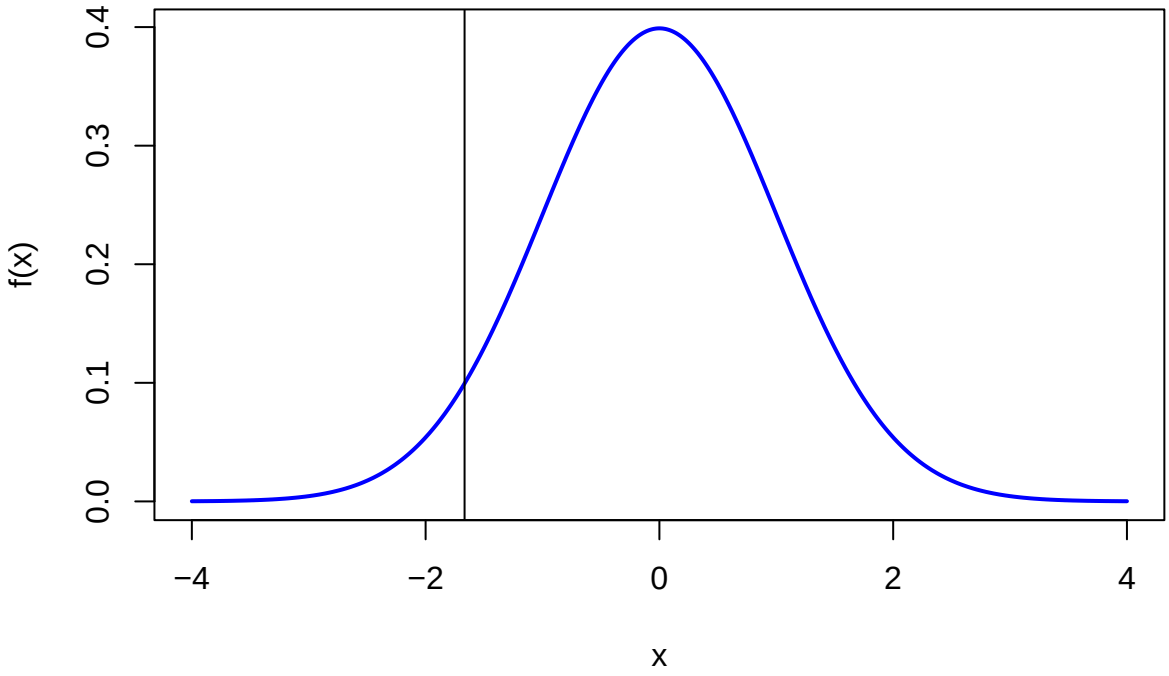
```
## [1] 0.0668072
```

The p-value is 0.0668072, which is greater than 0.05, so we accept the null hypothesis that $\mu_o = 6$.

Question 4

Assume that $n = 100$, but that $S = 6$ (a larger sample standard deviation, i.e., the individual data show 50% more variability in their values), with all other quantities being the same as in Question 1. What is your conclusion now?

```
n <- 100
z <- (5-6) / (6/sqrt(n))
fx = dnorm(seq(-4,4,by=0.01))
plot(seq(-4,4,by=0.01), fx, typ="l", col="blue", lwd=2, xlab="x", ylab="f(x)")
abline(v=z)
```




```
p <- pnorm(z)
p
```

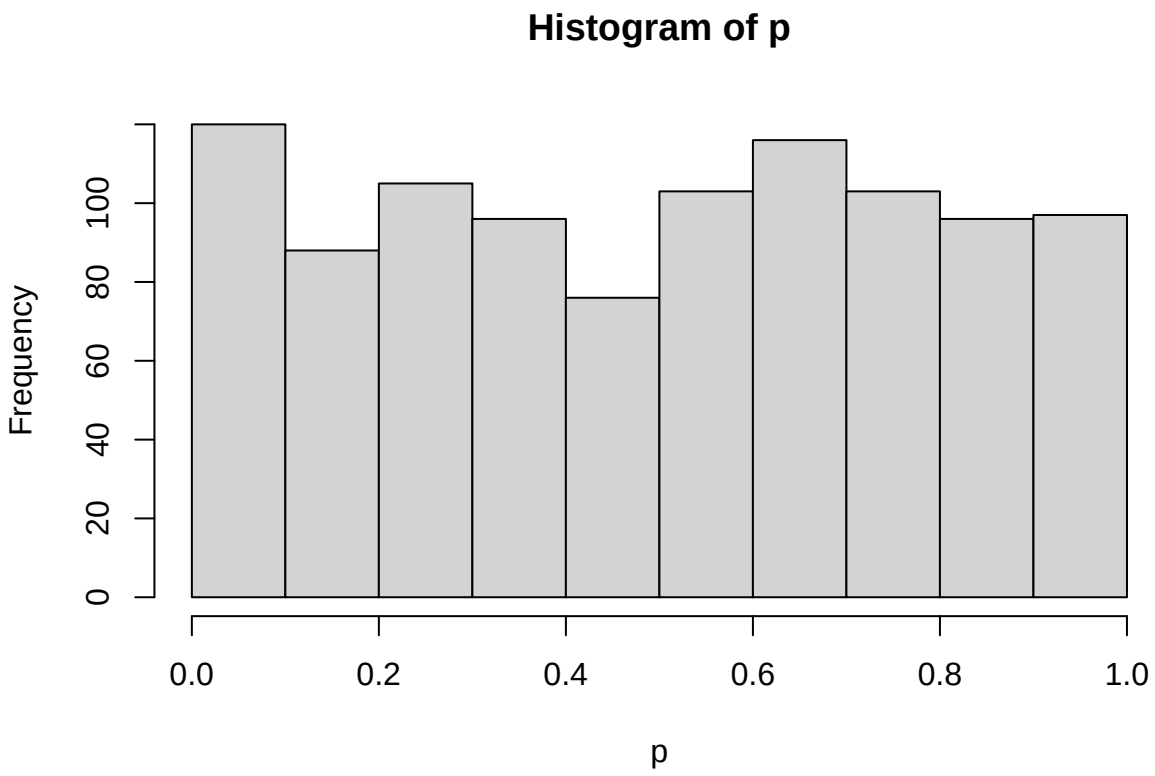
```
## [1] 0.04779035
```

The p-value is 0.04779035, which is less than 0.05, so we reject the null hypothesis that $\mu_o = 6$.

Question 5

In the notes, we mention that if the null is true, the distribution for the p-value is uniform, with values between 0 and 1. Let's demonstrate that here.

```
set.seed(404)
p <- rep(NA,1000)       # set aside storage for 1000 p-values
n <- 100                # assume a sample size of 100 for each experiment
for ( ii in 1:1000 ) { # do 1000 experiments
  x <- rnorm(n)         # sample 100 values from a standard normal
  z <- (mean(x)-0)/(sd(x)/sqrt(n))  # test a (true!) null mu_o = 0
  p[ii] <- pnorm(z)     # do a lower-tail test
}
hist(p)                # make a histogram of p-values
```



What do you observe? Is it plausible to conclude that when the null hypothesis is true, the p-values are distributed uniformly?

Yes, we can conclude that the p-values are distributed uniformly when the null hypothesis is true.

Question 6

In the notes, I said we'd come back to the Kolmogorov-Smirnov test later. Now is later.

Look at the documentation for the function `ks.test()`. You can do this by going to the console and typing `?ks.test` at the prompt, or by going to `RStudio`'s `Help` pane and searching for `ks.test`. Let's look at three arguments here:

- `x`: as we will do a one-sample KS test here, that would be the variable p from the last question.
- `y`: as we will test to see if the p-values are uniformly distributed, we will name a cdf function here... the example is `pnorm`, but here we want `punif`, as we are looking at the uniform distribution, not the normal.
- `alternative`: you need not specify this, as we will do a default two-sided test, but you should realize this is where you specify the range of alternative hypotheses.

Next, scroll down to `Value`. The `Value` section in the documentation lists what the function returns. As you can see, `ks.test()` returns a list with five different entries. The second listed one is `p.value`, which is what we want. In order to see what the p-value is, you can do something like this:

```
ks.test.out = ks.test(...enter arguments here...)
print(ks.test.out$p.value)
```

This is important! *To access a particular list element, type the name of the list followed by a dollar sign followed by the name of the element.* Foreshadowing: this is how you access the data in columns of a data frame... because a data frame is just a list where every element is a vector and every vector has the same length.

So, now: test whether the p-values derived in Question 5 are uniformly distributed. Print the p-value for the KS test and state a qualitative conclusion.

```
ks.test.out = ks.test(p, "punif")
ks.test.out$p.value
```

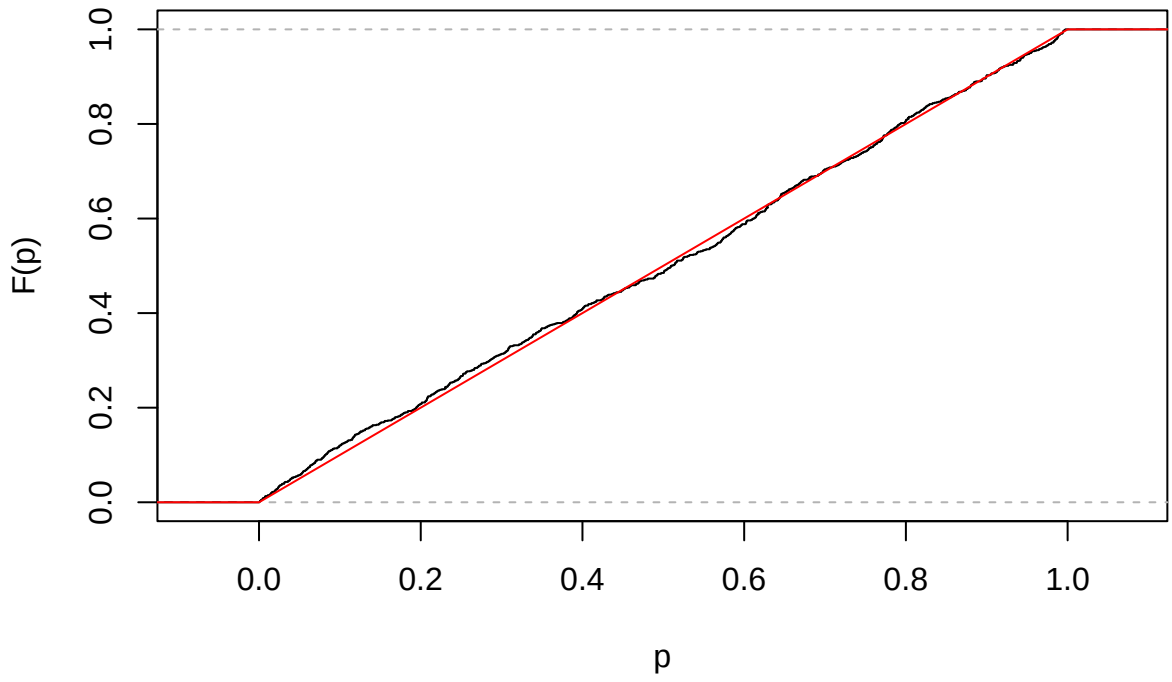
```
## [1] 0.591583
```

Conclusion: The p-value of Kolmogorov-Smirnov test is 0.591583, which is greater than 0.05, so we accept the null hypothesis. That means p-values are distributed uniformly.

You can find more information on the KS test on this web page.

As a parting note, let's look at what the KS test is actually doing.

```
plot(ecdf(p),xlab="p",ylab="F(p)",main=NULL)
lines(c(-1,0,1,2),c(0,0,1,1),col="red")
```



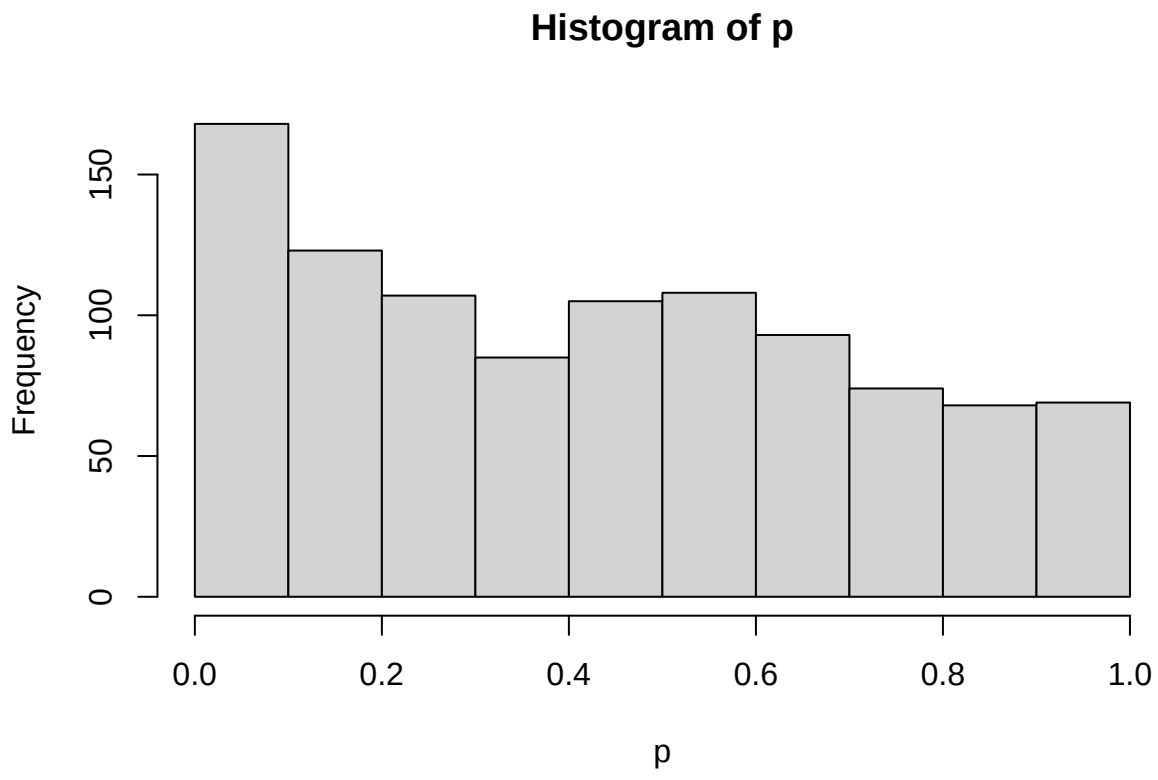
The KS test statistic is the maximum vertical distance from the red line (the cumulative distribution function for a Uniform(0,1) distribution) to the black curve (the empirical cumulative distribution function for the p-values). That distance is a random variable whose distribution, under the null, is the Kolmogorov distribution. If the observed maximum distance is sufficiently large, then we would reject the null hypothesis and

conclude that the p-values are *not* uniformly distributed. **Again, hypothesisi testing is all about a test statstic (computed from the data), its assumed distribution under the null, and determining the probability of observing the test statistic or a value more extreme (which is the p-value). Everything else is test-specific detail.**

Question 7

Repeat the analysis in Questions 5 and 6, but have your null hypothesis be that $\mu_o = 0.025$. (This means replace the 0 in the code in Question 5 with 0.025.) Cut-and-paste code such that you simulate the p-value data, then create the histogram of p-values and run the KS test. What do you conclude now? Are the p-values uniformly distributed?

```
set.seed(404)
p <- rep(NA,1000)
n <- 100
for ( ii in 1:1000 ) {
  x <- rnorm(n)
  z <- (mean(x)-0.025)/(sd(x)/sqrt(n))
  p[ii] <- pnorm(z)
}
hist(p)
```




```
ks.test.out = ks.test(p, "punif")
ks.test.out$p.value
```

```
## [1] 5.428641e-10
```

Conclusion: The p-value of KS test is 5.428641e-10, which is less than 0.05, so we reject the null hypothesis. That means p-values are not distributed uniformly.

Stop and think for a second about what you are observing: as the truth and the assumed null diverge, observing a p-value with value $< \alpha$ becomes more and more likely. (This is assuming that the value of the truth is consistent with what is being tested, e.g., the true value gets smaller and smaller for a lower-tail test, or larger and larger for an upper-tail test. Otherwise the p-values start skewing the other way, towards 1.)

Question 8

Take your p-values from Question 7 and determine the percentage that are less than 0.05. (Reminder: relational operator, `sum()`, `length()`, etc.) You should see that this percentage is $> 5\%$!

```
pct <- length(which(p < 0.05)) / length(p) *100
pct
```

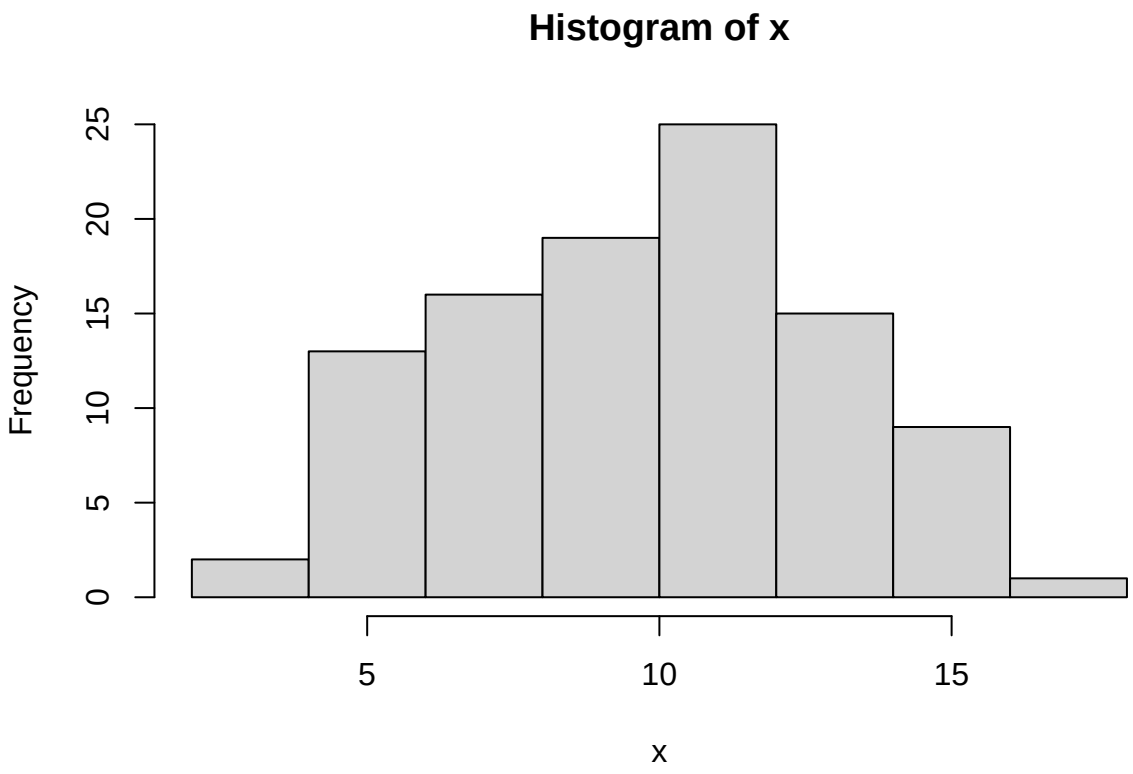
```
## [1] 9.7
```

Question 9

Now let's transition to the t test. To remind you, a one-sample t test takes a sample of independent and identically (and normally) distributed data and tests for a hypothesized mean.

Here is some data:

```
set.seed(606)
x <- rnorm(100,mean=10,sd=3)
hist(x)
```



We will use `t.test()` to perform a lower-tail test with hypothesized mean $\mu_o = 10.5$.

Look at the help page for `t.test()`. What you will see is a similar function to `ks.test()`: you have the arguments `x`, `y`, and `alternative`, and in addition `mu`. You will leave `y` out of the function call, since we are doing a one-sample test. For `mu`, specify 10.5. Go ahead and code the test, and state a conclusion.

```
t.test(x, mu=10.5)
```

```
##
##  One Sample t-test
##
## data:  x
## t = -2.1171, df = 99, p-value = 0.03676
## alternative hypothesis: true mean is not equal to 10.5
## 95 percent confidence interval:
##   9.179013 10.457208
## sample estimates:
## mean of x
##  9.818111
```

Conclusion: The p-value of t-test is 0.03676, which is less than 0.05, so we reject the null hypothesis. That means the mean of data is not 10.5, but 9.818111.

Note that your output will look something like this (but not exactly like this, because I changed the numbers):

```
    One Sample t-test

data:  x
t = -1.3409, df = 99, p-value = 0.09151
alternative hypothesis: true mean is less than 10.25
95 percent confidence interval:
     -Inf 10.35291
sample estimates:
mean of x
 9.818111
```

At the bottom is $\bar{X}$. The second line of output shows the standardized test statistic ($t = -1.3409$), the number of degrees of freedom ($\nu = n-1 = 99$), and the p-value (here, 0.09...assuming $\alpha = 0.05$, we conclude that we fail to reject the null). The last bit to point out is the "95 percent confidence interval."

We haven't talked about confidence intervals yet, and I don't really intend to, because confidence intervals and hypothesis tests are "inverses" of one another: you can transform a hypothesis test into a statement of a confidence interval and vice-versa. It suffices to say two things here:

- A confidence interval is a random interval that has a $1-\alpha$ percent chance (here, 95%) of overlapping the true value of the mean.
- If a $1-\alpha$ confidence interval overlaps μ_o (not the truth, but μ_o), then $p > \alpha$, i.e., you will fail to reject the null hypothesis.

If you are dealing with confidence intervals on your own and need guidance, talk to me or a TA.

Question 10

Let's add a new data sample:

```
set.seed(303)
y <- rnorm(50,mean=10.5,sd=3.5)
```

Perform a two-sided two-sample t test with x and y as input and state a conclusion. Assume that the null hypothesis for the difference in the means is zero. (Also assume that you do not know the true means and thus that the alternative is that $\mu_x \neq \mu_y$.)

```
t.test(x, y)
```

```
##
##  Welch Two Sample t-test
##
## data:  x and y
## t = -1.532, df = 99.705, p-value = 0.1287
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.9382290  0.2491989
## sample estimates:
## mean of x mean of y
##  9.818111 10.662626
```

Conclusion: The p-value of t-test is 0.1287, which is greater than 0.05, so we accept the null hypothesis. We can say that the true difference in means is zero.

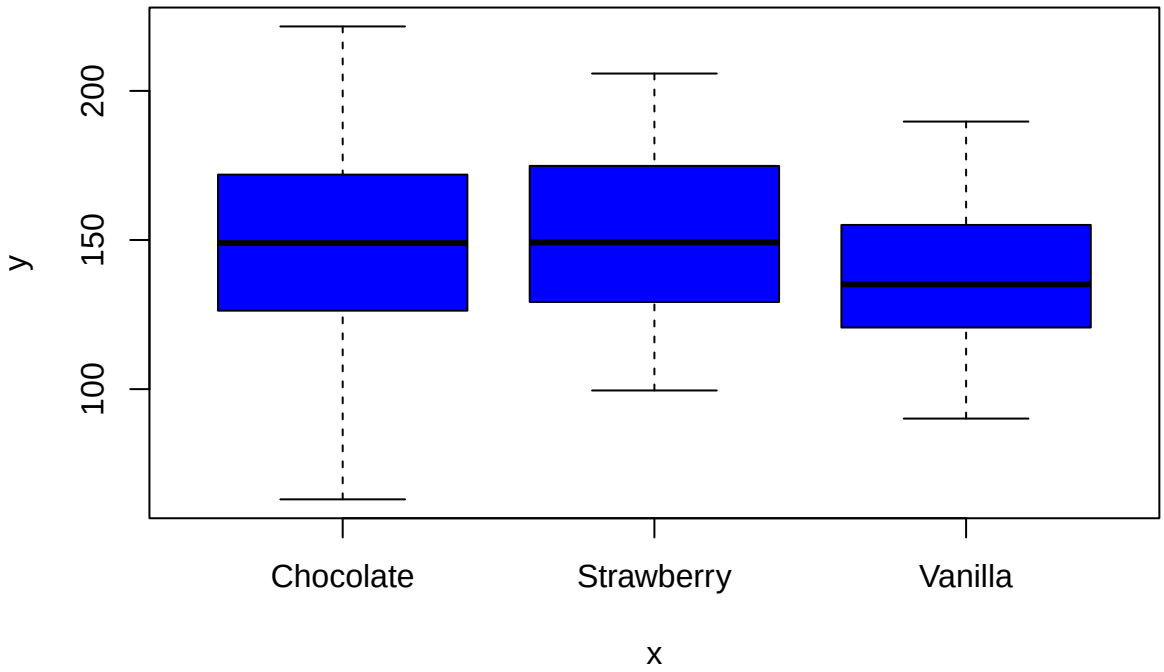
Question 11

And now, one-way analysis of variance, or one-way ANOVA. Here I will create a fake dataset in which each person states his or her preferred ice cream flavor and his or her weight:

```
x = c(rep("Chocolate",40),rep("Vanilla",30),rep("Strawberry",30))
x = factor(x)
y = c(rnorm(40,mean=145,sd=30),rnorm(30,mean=140,sd=25),rnorm(30,mean=148,sd=32))
```

Let's visualize these data with a boxplot, grouping on the factor variable `x`:

```
boxplot(y~x,col="blue")
```



Visually, we see that vanilla ice cream enthusiasts tend to have lower weights. But is this observation actually statistically significant? One-way ANOVA can help answer that question. The "one-way" refers to the fact that in this experiment, there is one factor variable, or "treatment": the ice cream flavor. One-way ANOVA tests the null hypothesis that the true means for each treatment group are the same, versus the alternative that *at least one* of the means is different from the others. (So one-way ANOVA will not tell us specifically that the vanilla treatment has the different mean, just that one of the means is different... that is, of course, if $p < 0.05$.)

One way to run a one-way ANOVA in `R` is to embed a call to the function `lm()` ("linear model") inside a call to `anova()`. Let's look at the documentation for `lm()`:

```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE,
   singular.ok = TRUE, contrasts = NULL, offset, ...)
```

None of this matters to us now, aside from `formula`. A model formula puts the dependent, or response variable on the left side of a tilde, and the independent, or predictor variables on the right side. Here, there is just one dependent variable and one response variable, so we have a simple formula:

```
y~x
```

(You saw this formula already, in the call to `boxplot()` above.) This formula is the only thing you have to specify in the call to `lm()` here. (Other variables like `data` look like they have to be specified, but, no, they are optional. This is an example of `R`'s imperfect/inconsistent documentation. Get used to it.)

So: call `anova()` and state a conclusion: is at least one of the mean weights different from the others?

```
anova(lm(y~x))
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value Pr(>F)
## x          2   4131 2065.73  2.2936 0.1064
## Residuals 97  87364  900.65
```

Conclusion: the p-value of the F-statistic is 0.1064, which is greater than 0.05, so we accept the null hypothesis that the true means for each treatment group are the same.

Here's an example of the output that you would see from `anova()`:

```
Analysis of Variance Table

Response: y
          Df Sum Sq Mean Sq F value Pr(>F)
x          2    993  496.48  0.8861 0.4156
Residuals 97  54348  560.29
```

The p-value is to the upper right (`Pr(>F)`). The basic idea is we compute the sum of the squared differences between each mean and the global mean value (here, 993, with the `Df` of 2 saying that we have 3 treatment groups), and divide one value by the other: $993/2 = 496.48$. The second row shows a similar calculation, but the differences are between each datum and its group mean. The average value there is 560.29. The ratio of 494.48 to 560.29 gives the F statistic (0.8861), and the p-value is computed assuming an F distribution for 2 numerator and 97 denominator degrees of freedom.

This details are all fine and dandy, but the basic idea is if the variability *within* each group is small (small `Mean Sq` in the second row) but the variability *between* groups is large (large `Mean Sq` in the first row), then F will be large, the p-value will be small, and we'll detect that there's a difference between the group means.