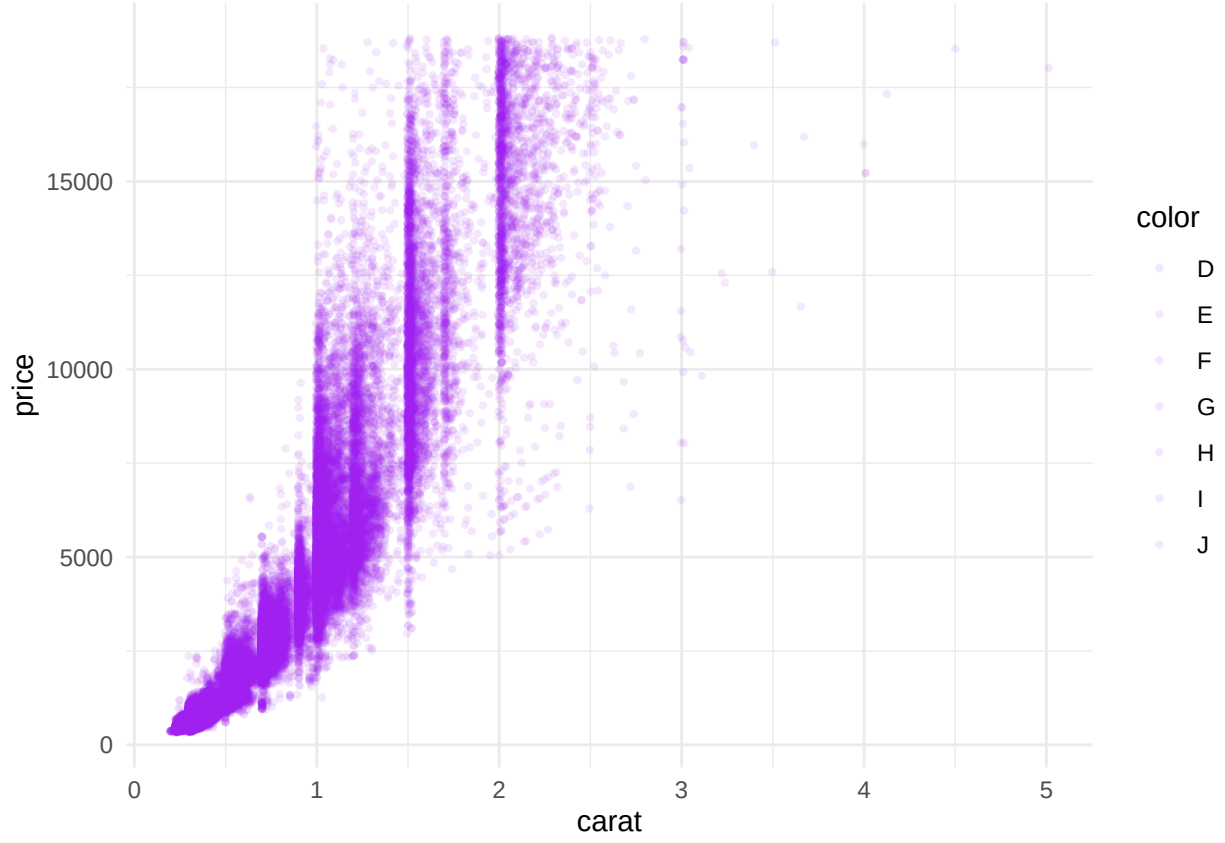
Untitled

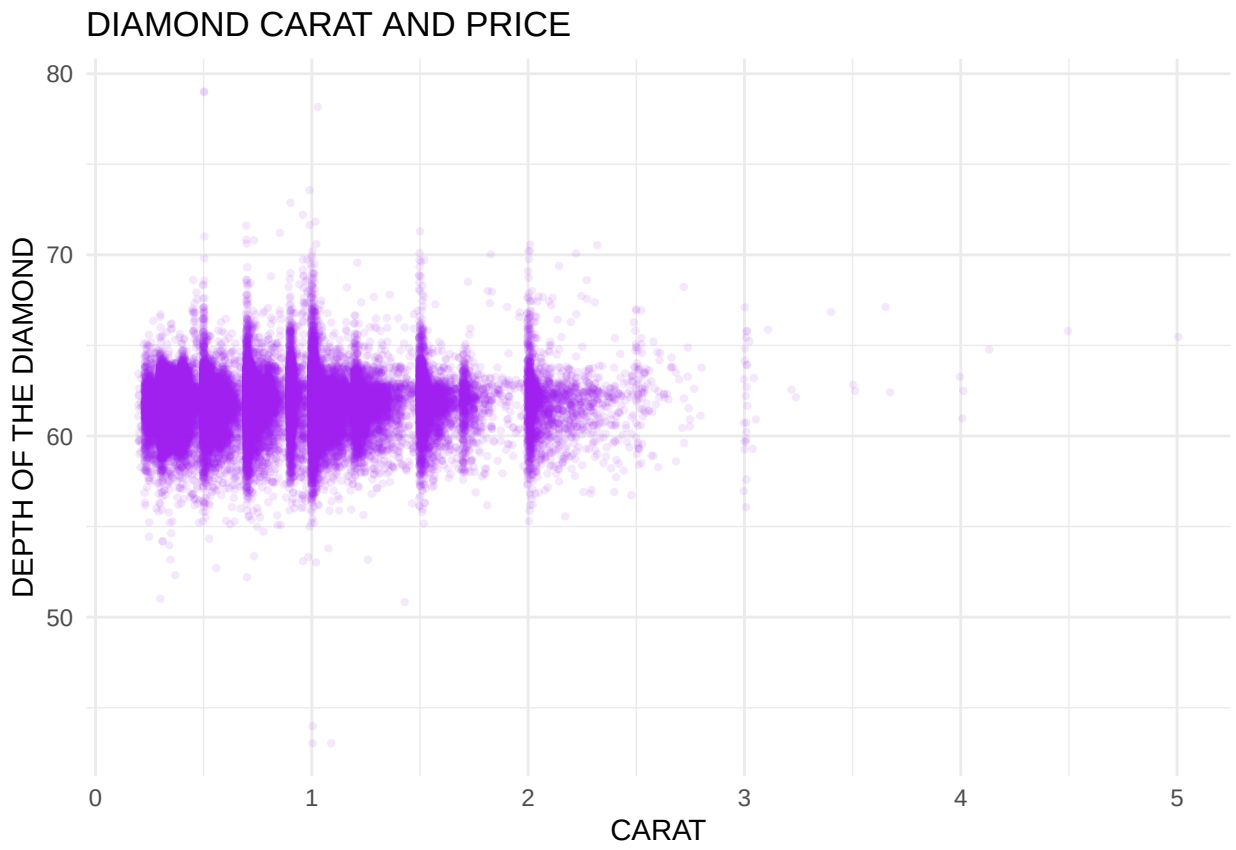
2025-08-06

```
pacman :: p_load("ggplot2", "dplyr" )

View(diamonds)

# Q 1
ggplot(diamonds, mapping = aes(x = carat, y = price, fill = color)) +
  geom_jitter(alpha = 0.1 , size = 0.8, color = "purple") +
  theme_minimal()
```

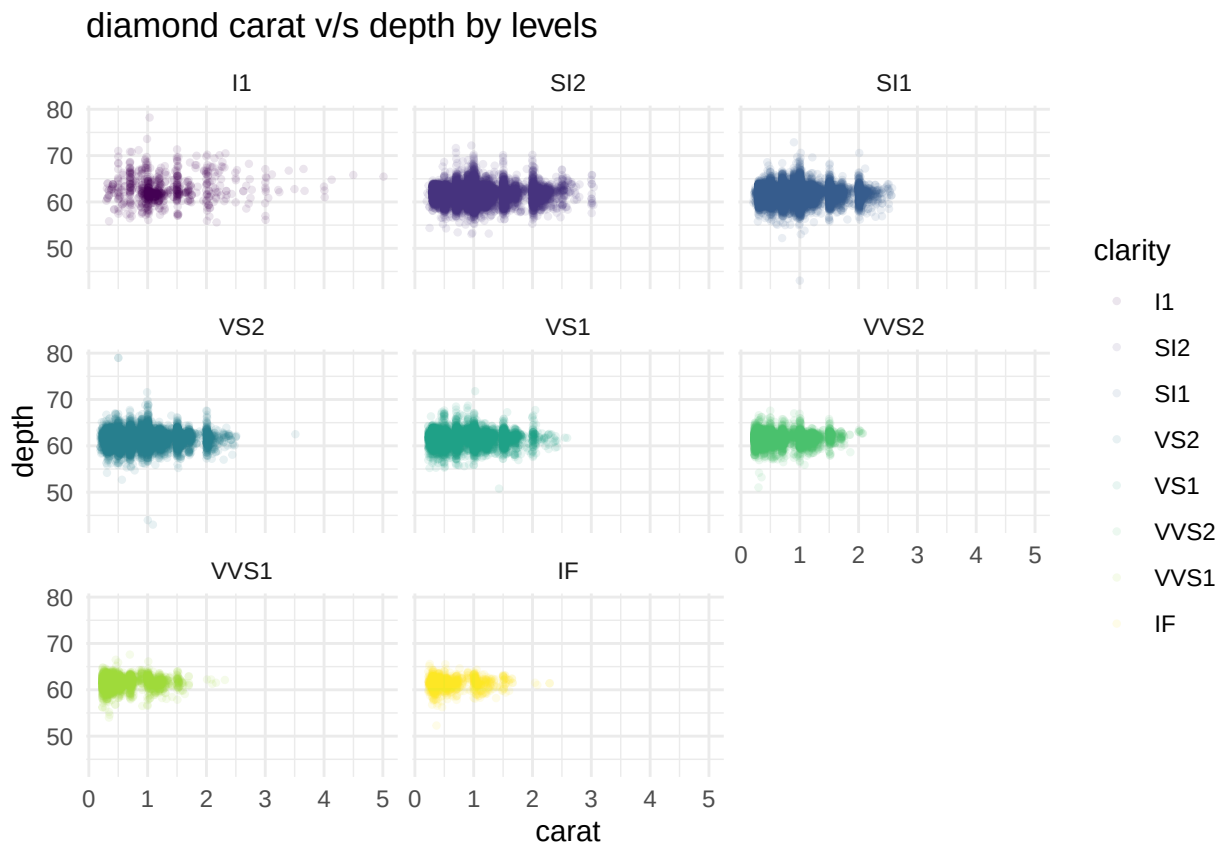



```
# Q 2

ggplot(diamonds, mapping = aes(x = carat, y = depth , color = clarity)) +
  geom_jitter(alpha = 0.1, size = 0.8, color = "purple") +
  labs( x = "CARAT",
        y = "DEPTH OF THE DIAMOND",
```

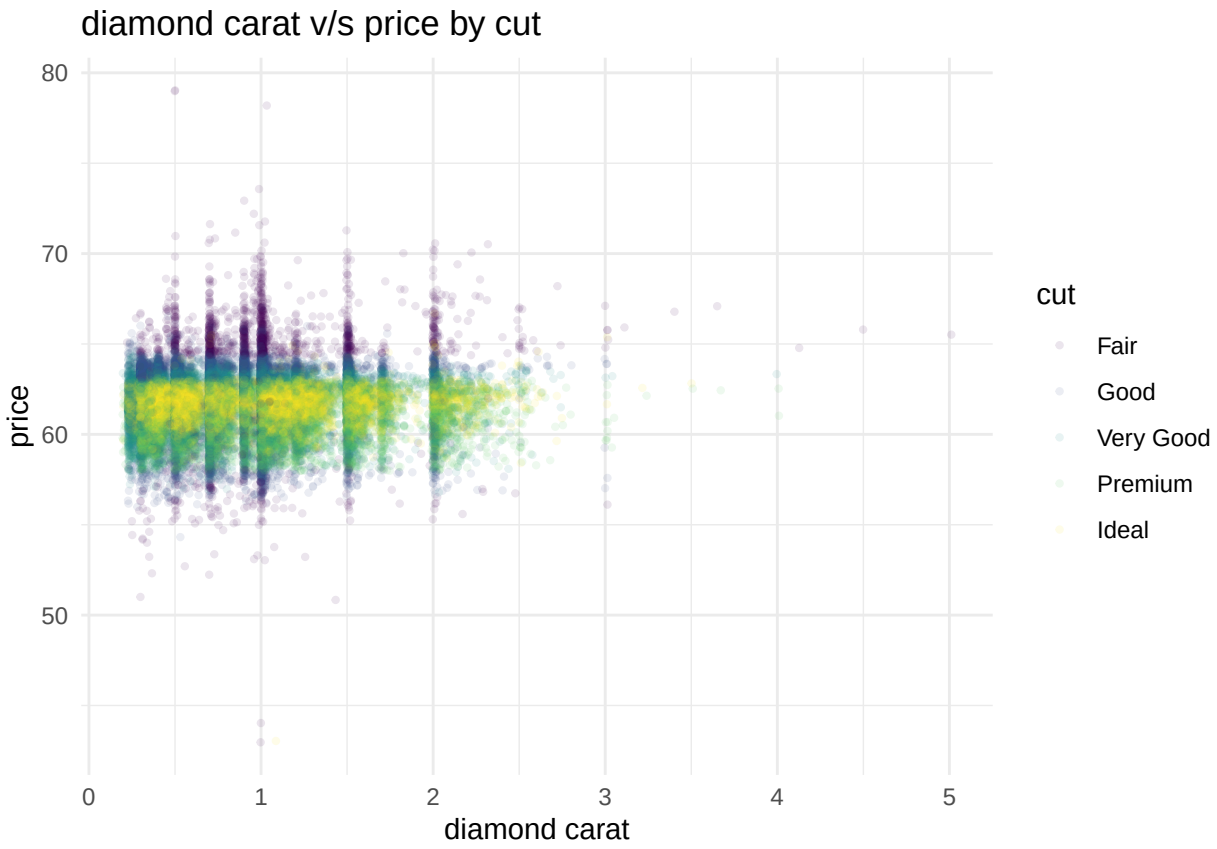
```
       title = "DIAMOND CARAT AND PRICE") +
  theme_minimal()
```



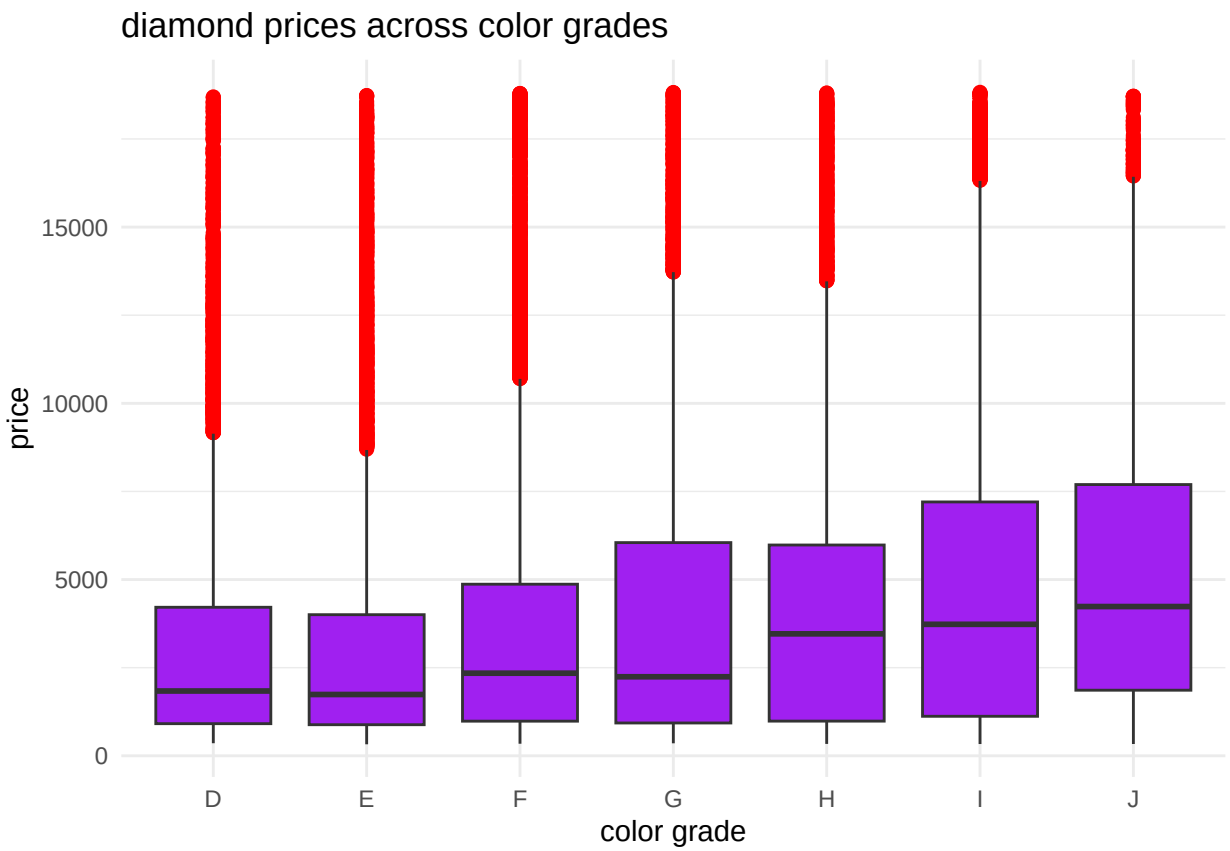

```
# Q 3

ggplot(diamonds, aes(x = carat, y = depth, color = clarity)) +
   geom_jitter(alpha = 0.1 , size = 0.8) +
  theme_minimal() +
  facet_wrap( ~ clarity) +
  labs( title = "diamond carat v/s depth by levels",
        x = "carat",
        y = "depth")
```



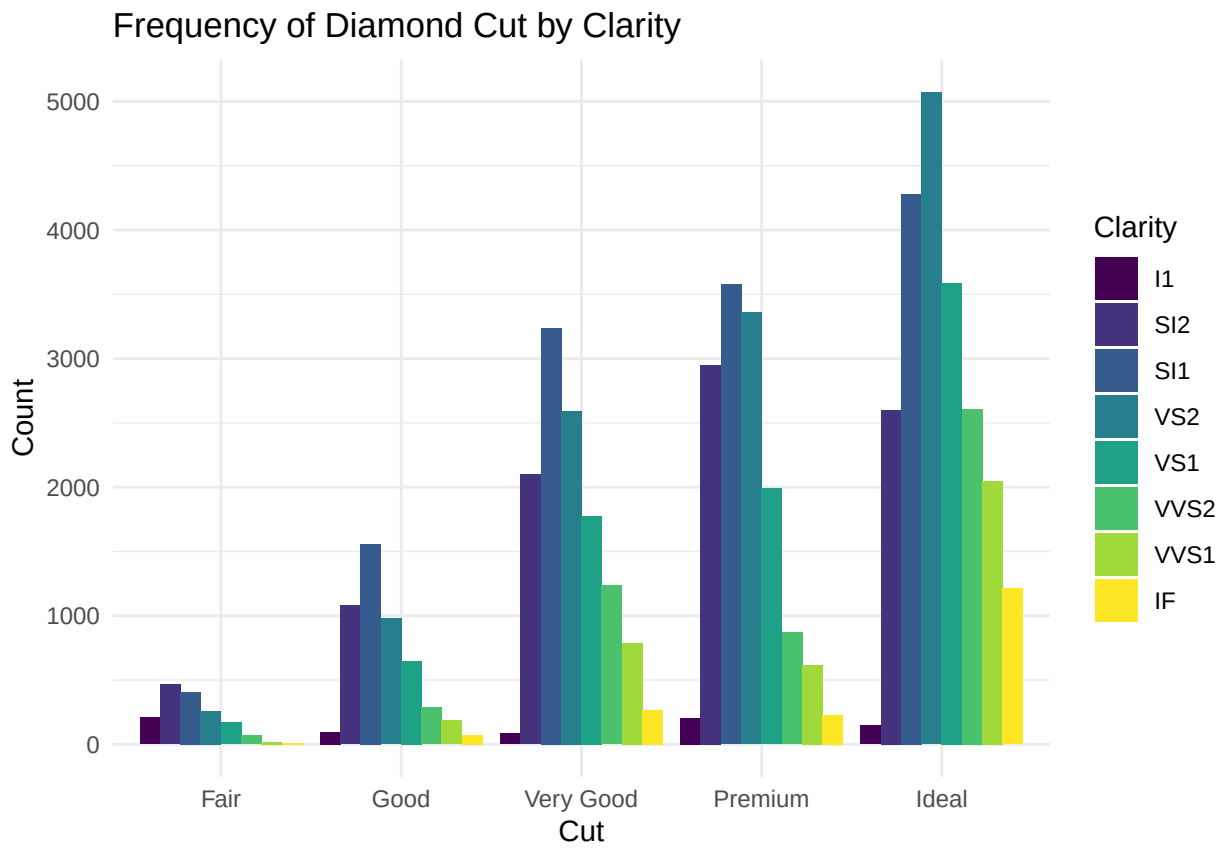

```
# Q 4

ggplot(diamonds, aes(x = carat, y = depth, color = cut)) +
  geom_jitter(alpha = 0.1, size = 0.8)+
  theme_minimal() +
  labs(title = "diamond carat v/s price by cut",
       x = "diamond carat",
       y = "price",
       color = "cut")
```



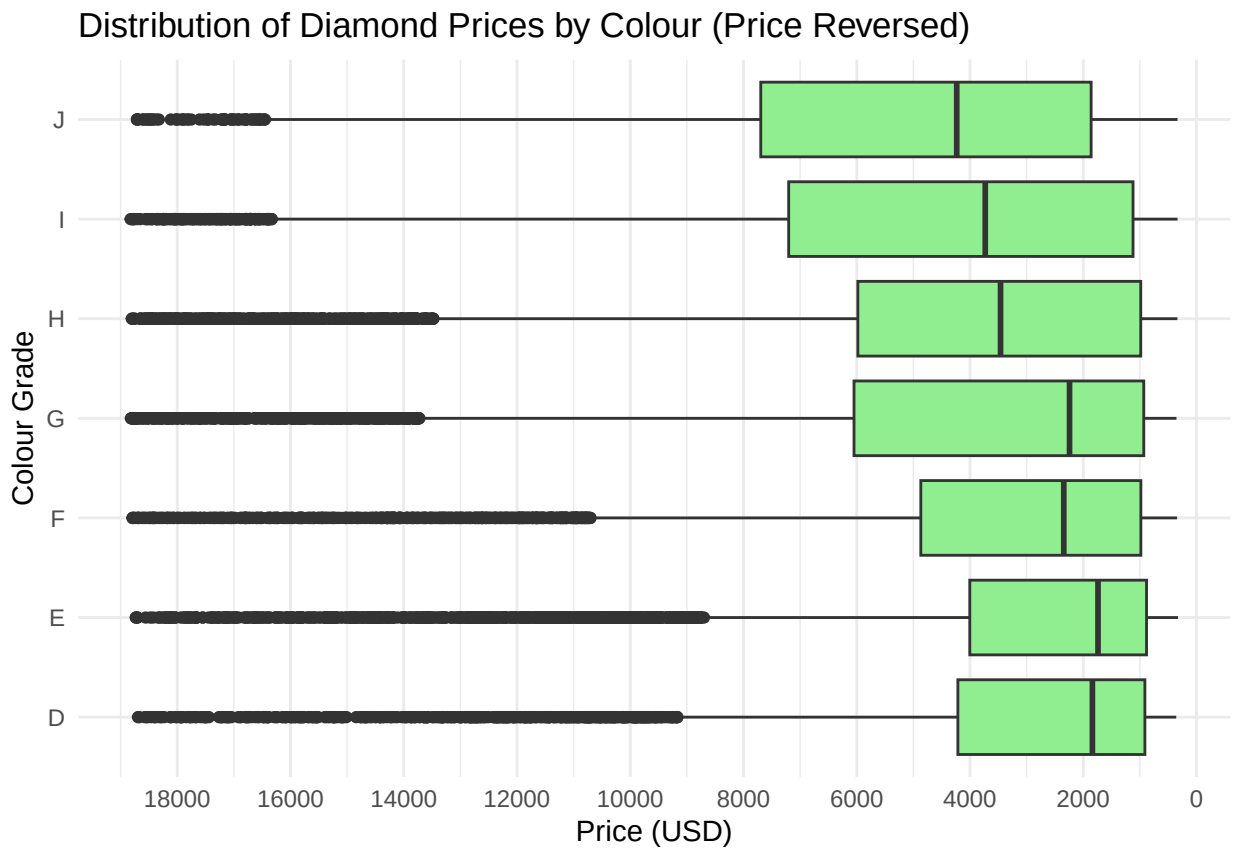

```
# Q 5

ggplot(diamonds, aes(x = color , y = price)) +
  geom_boxplot(outlier.colour  = "red", outlier.size = 2, fill  = "purple") +
  theme_minimal() +
  labs(title = "diamond prices across color grades",
       x = "color grade",
       y = "price")
```



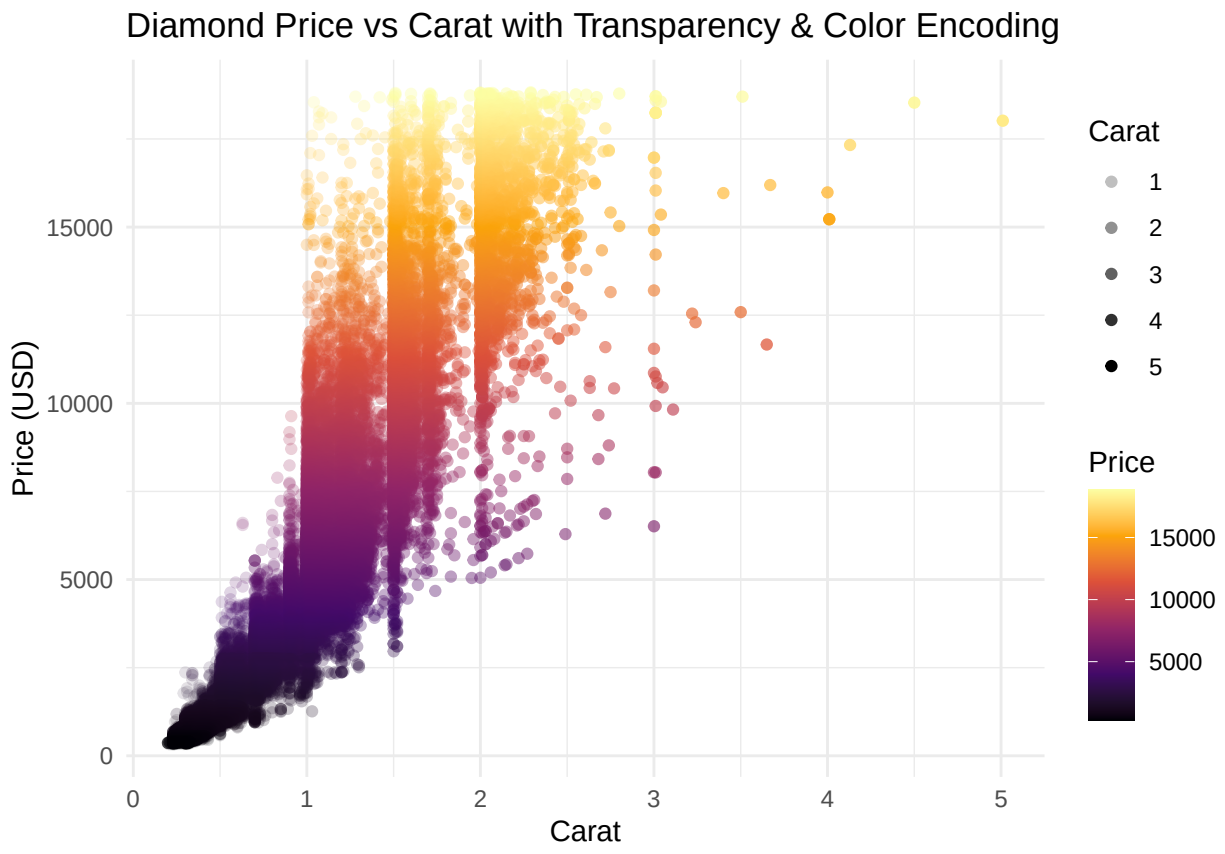

```
# Q 6

ggplot(diamonds, aes(x = cut, fill = clarity)) +
  geom_bar(position = position_dodge()) +
  theme_minimal() +
  labs(title = "Frequency of Diamond Cut by Clarity",
       x = "Cut",
       y = "Count",
       fill = "Clarity")
```



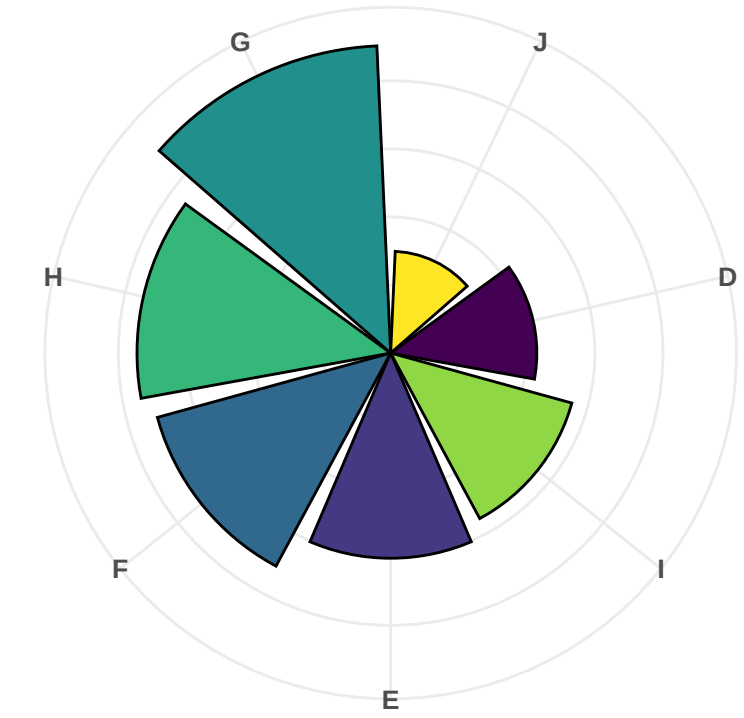

```
# Q 7

ggplot(diamonds, aes(x = color, y = price)) +
  geom_boxplot(fill = "lightgreen") +
  coord_flip() +
  scale_y_reverse(breaks = seq(0, 20000, by = 2000)) +
  theme_minimal() +
  labs(title = "Distribution of Diamond Prices by Colour (Price Reversed)",
       x = "Colour Grade",
       y = "Price (USD)")
```




```
# Q 8

ggplot(diamonds, aes(x = carat, y = price, alpha = carat, color = price)) +
  geom_point() +
  scale_alpha(range = c(0.1, 1)) +
  scale_color_viridis_c(option = "inferno") +
  theme_minimal() +
  labs(title = "Diamond Price vs Carat with Transparency & Color Encoding",
       x = "Carat",
       y = "Price (USD)",
       color = "Price",
       alpha = "Carat")
```



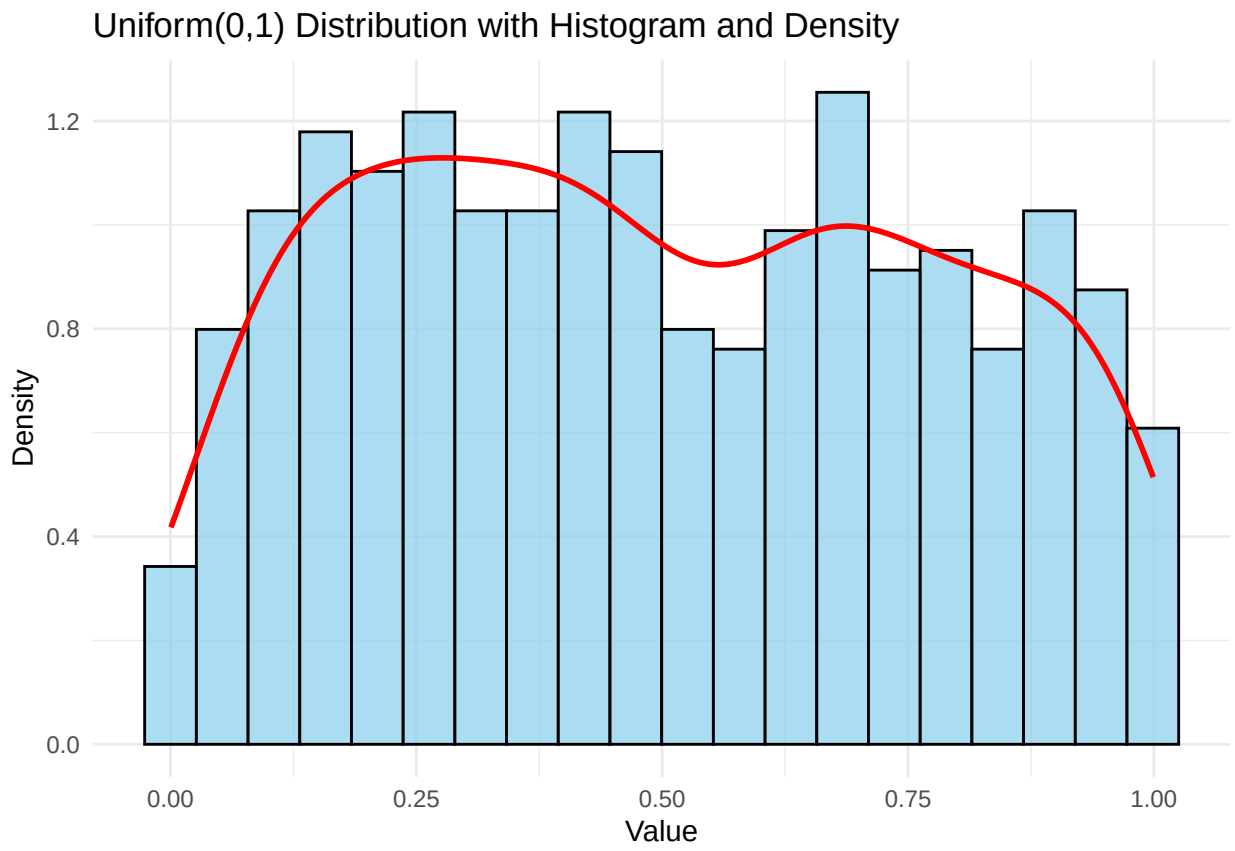

```
# Q 9


library(dplyr)

price_by_colour <- diamonds %>%
  group_by(color) %>%
  summarise(total_price = sum(price)) %>%
  arrange(total_price)

price_by_colour$angle <- 90 - 360 * (seq_along(price_by_colour$color) - 0.5) / nrow(price_by_colour)

ggplot(price_by_colour, aes(x = factor(color, levels = color), y = total_price, fill = color)) +
  geom_bar(stat = "identity", color = "black") +
  coord_polar(start = 0) +
  theme_minimal() +
  theme(axis.title = element_blank(),
        axis.text.y = element_blank(),
        axis.text.x = element_text(size = 10, face = "bold")) +
  labs(title = "Total Diamond Price by Colour Grade") +
  guides(fill = "none")
```



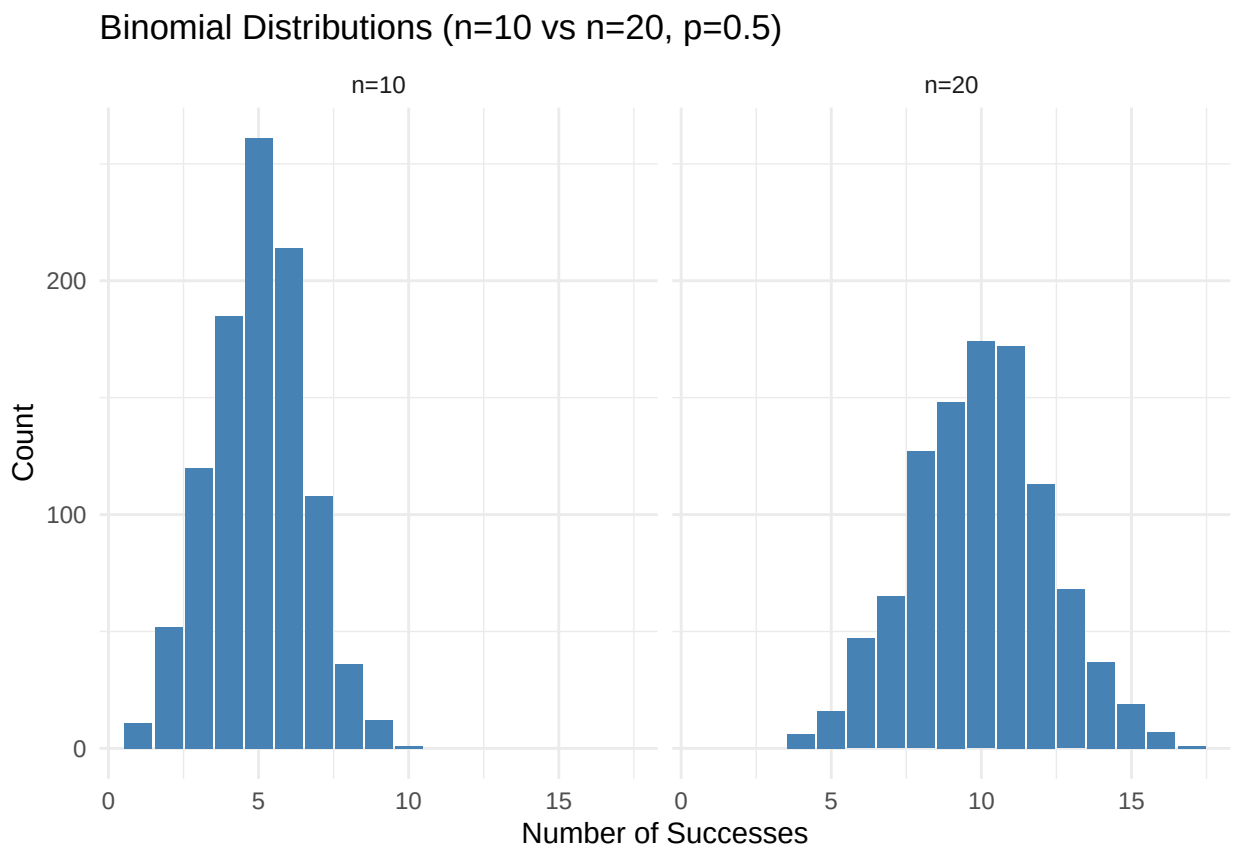

```
# 1

set.seed(123)
uniform_data <- data.frame(x = runif(500, 0, 1))

ggplot(uniform_data, aes(x = x)) +
  geom_histogram(aes(y = ..density..), bins = 20, fill = "skyblue", color = "black", alpha = 0.7) +
  geom_density(color = "red", size = 1) +
  theme_minimal() +
  labs(title = "Uniform(0,1) Distribution with Histogram and Density",
       x = "Value",
       y = "Density")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

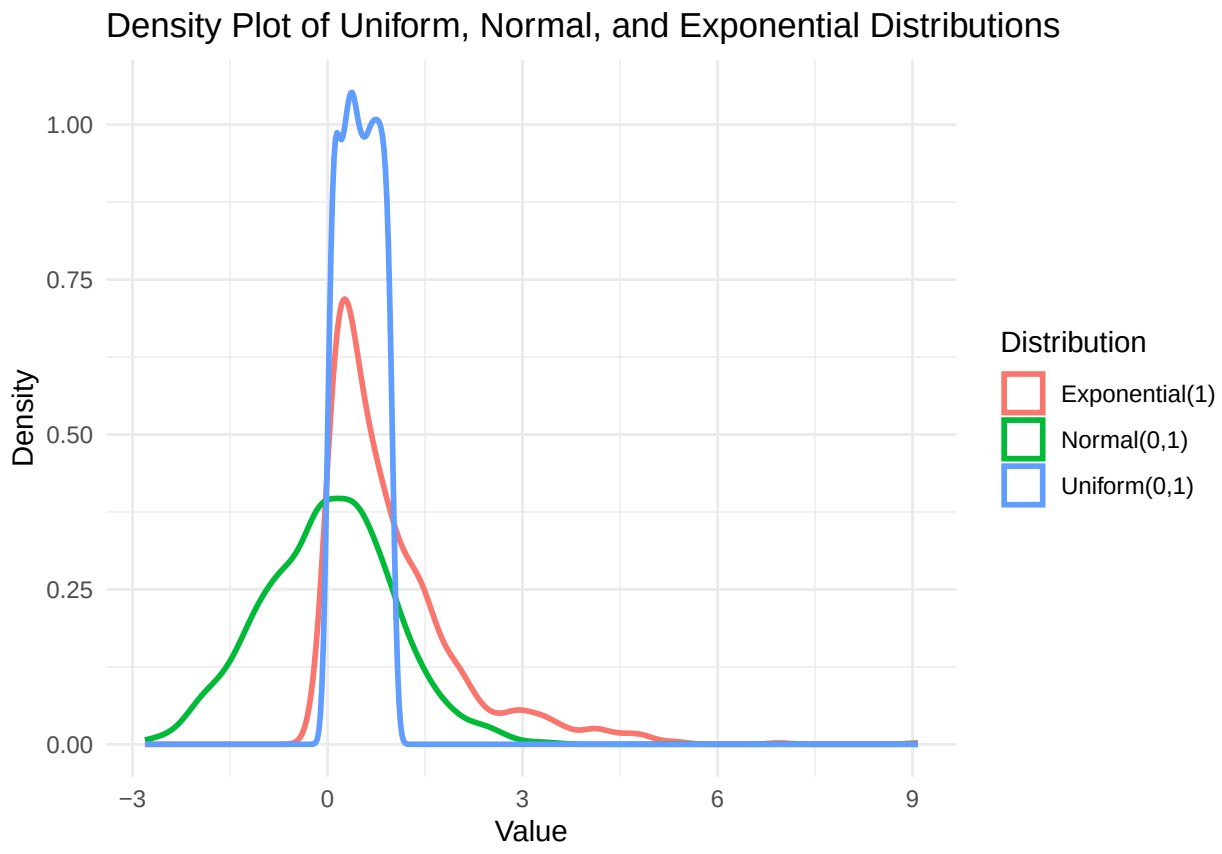
```
## Warning: The dot-dot notation (‘..density..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(density)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



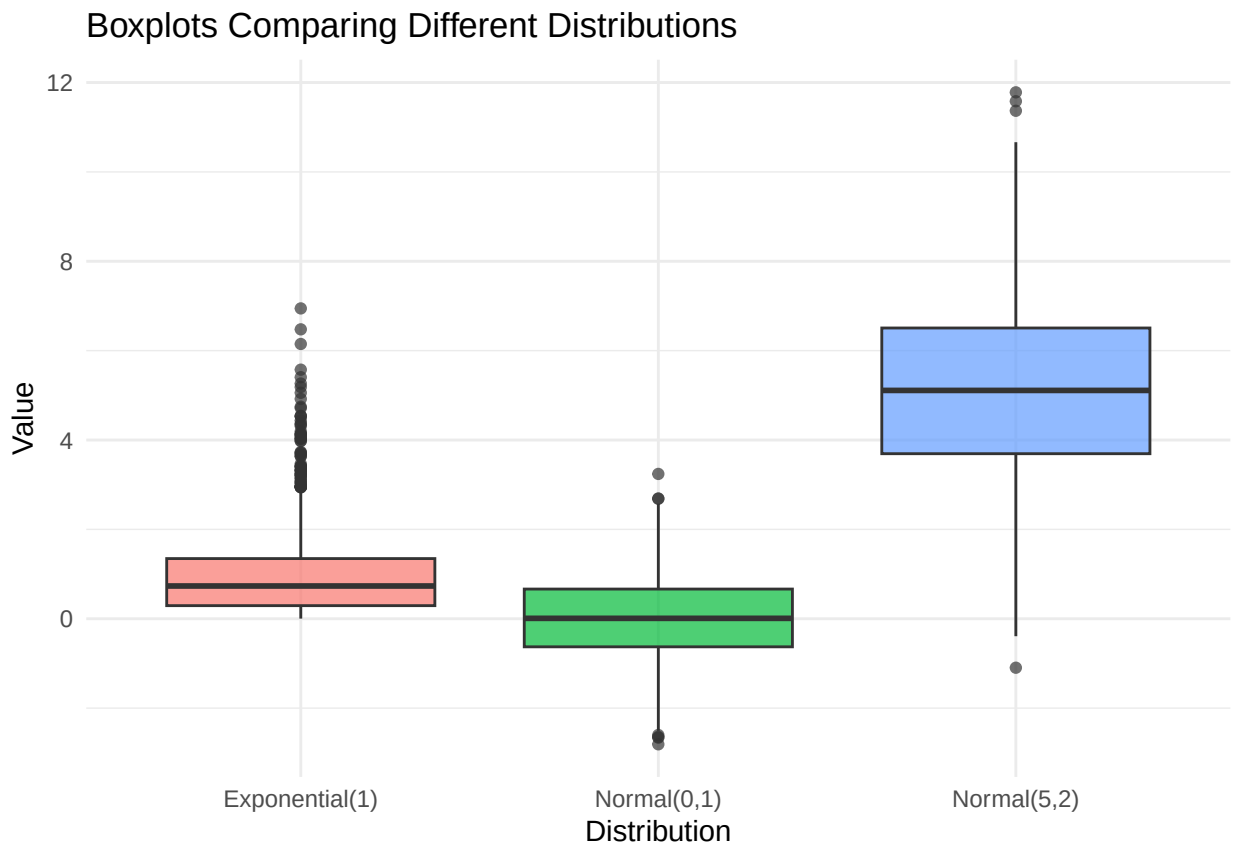

```r
# 2

binom_data <- data.frame(
  x = c(rbinom(1000, size = 10, prob = 0.5), rbinom(1000, size = 20, prob = 0.5)),
  n = factor(rep(c("n=10", "n=20"), each = 1000))
)

ggplot(binom_data, aes(x = x)) +
  geom_bar(fill = "steelblue") +
  facet_wrap(~ n) +
  theme_minimal() +
  labs(title = "Binomial Distributions (n=10 vs n=20, p=0.5)",
       x = "Number of Successes",
       y = "Count")
```



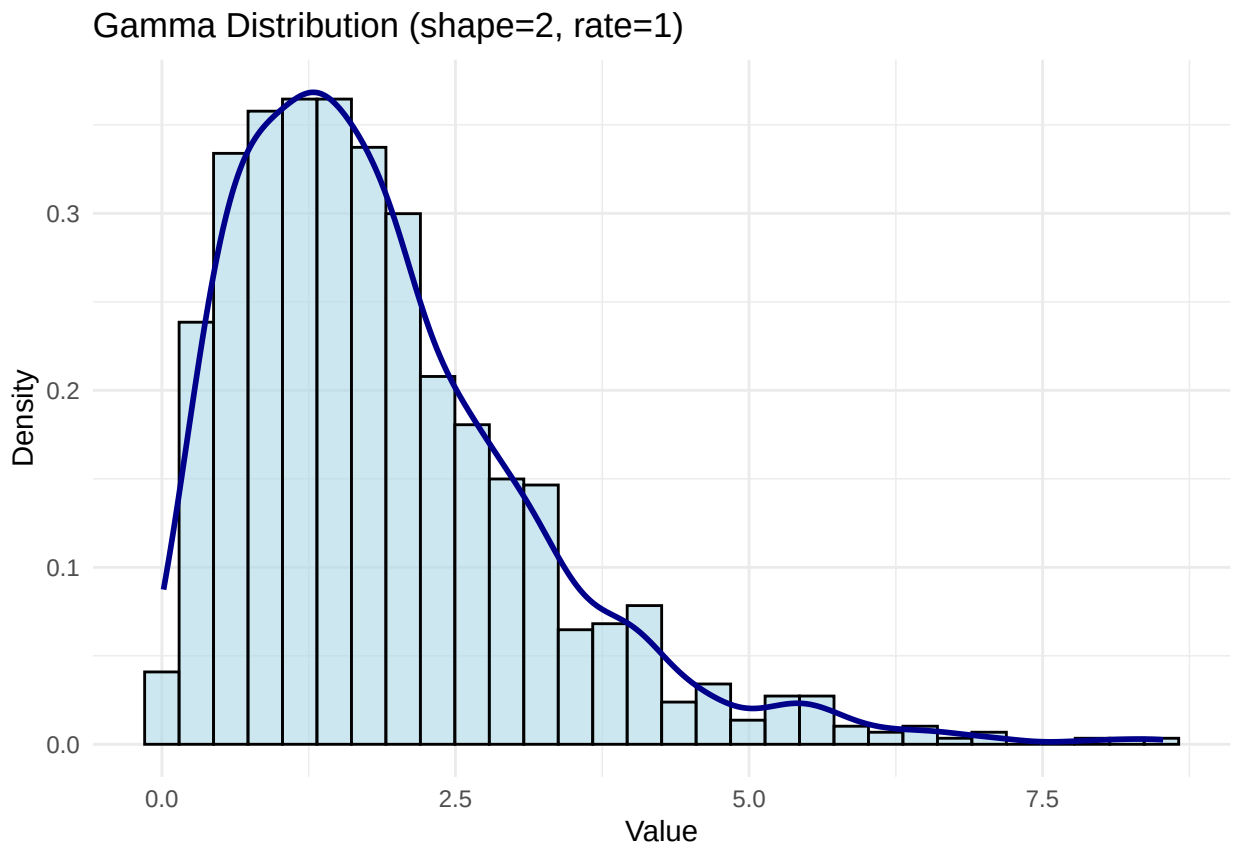

```r
# 3

set.seed(123)
data_unif <- data.frame(x = runif(1000), dist = "Uniform(0,1)")
data_norm <- data.frame(x = rnorm(1000), dist = "Normal(0,1)")
data_exp <- data.frame(x = rexp(1000, rate = 1), dist = "Exponential(1)")

all_data <- rbind(data_unif, data_norm, data_exp)

ggplot(all_data, aes(x = x, color = dist)) +
  geom_density(size = 1) +
  theme_minimal() +
  labs(title = "Density Plot of Uniform, Normal, and Exponential Distributions",
       x = "Value",
       y = "Density",
       color = "Distribution")
```



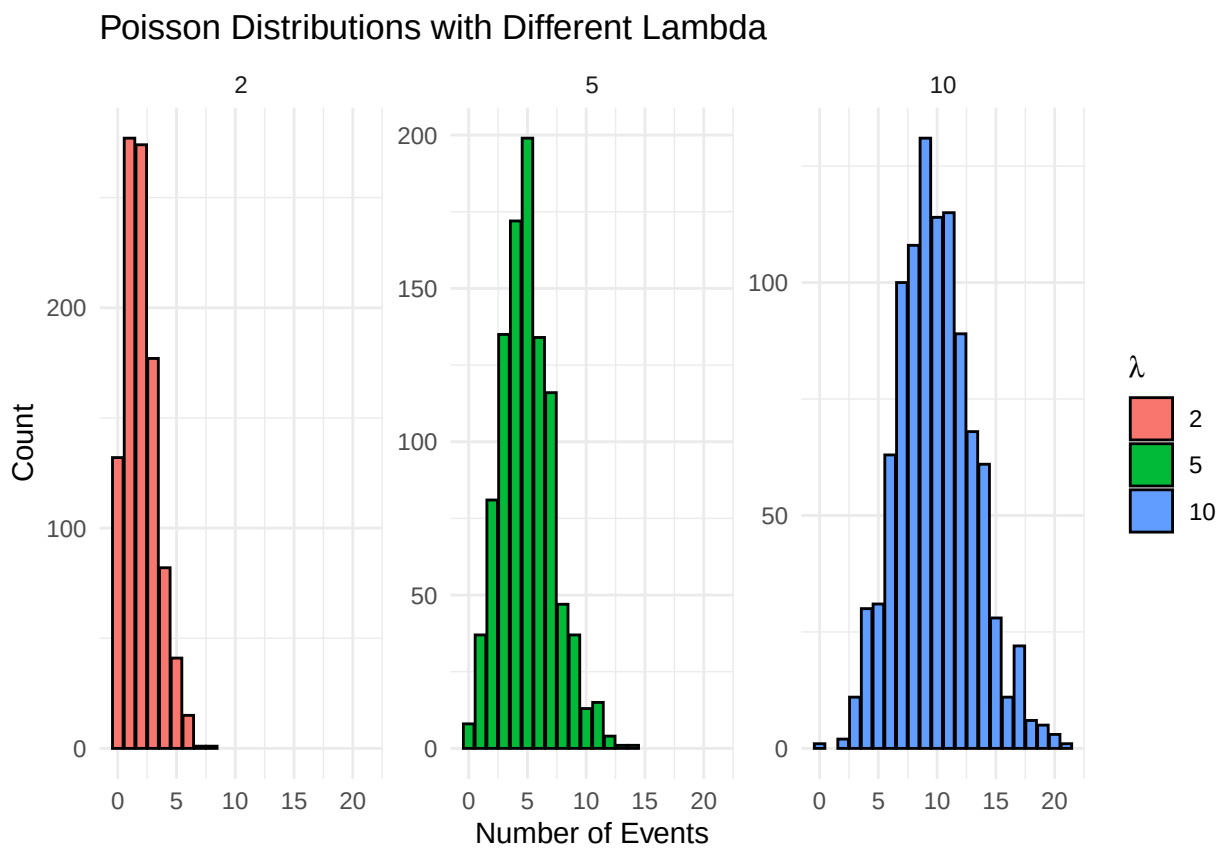

```r
# 4

set.seed(123)
box_data <- data.frame(
  value = c(rnorm(1000, 0, 1), rnorm(1000, 5, 2), rexp(1000, 1)),
  dist = factor(rep(c("Normal(0,1)", "Normal(5,2)", "Exponential(1)"), each = 1000))
)

ggplot(box_data, aes(x = dist, y = value, fill = dist)) +
  geom_boxplot(alpha = 0.7) +
  theme_minimal() +
  labs(title = "Boxplots Comparing Different Distributions",
       x = "Distribution",
       y = "Value") +
  theme(legend.position = "none")
```




```r
# 5

set.seed(123)
gamma_data <- data.frame(x = rgamma(1000, shape = 2, rate = 1))

ggplot(gamma_data, aes(x = x)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightblue", color = "black", alpha = 0.6) +
  geom_density(color = "darkblue", size = 1) +
  theme_minimal() +
  labs(title = "Gamma Distribution (shape=2, rate=1)",
       x = "Value",
       y = "Density")
```




```r
# 6

set.seed(123)
pois_data <- data.frame(
  x = c(rpois(1000, 2), rpois(1000, 5), rpois(1000, 10)),
  lambda = factor(rep(c(2, 5, 10), each = 1000))
)

ggplot(pois_data, aes(x = x, fill = lambda)) +
  geom_bar(position = "dodge", color = "black") +
  facet_wrap(~ lambda, scales = "free_y") +
  theme_minimal() +
  labs(title = "Poisson Distributions with Different Lambda",
       x = "Number of Events",
       y = "Count",
       fill = expression(lambda))
```




```
## FOR YOU, FOR ALL OF US
```