My son (6 yrs) and I tried out the pop the balloons game yesterday. Write the numbers 2-12, colour in some balloons next to some of the numbers and then pop them by throwing two dice that sum to the balloon number. Here are our score sheets.

Score sheets for our balloon popping by dice throw.

We then added our tallies together to see how many of each roll we got. I asked where he'd put his balloons next time to pop them faster (4, 6, 8, 10, 12), I don't think he was making the link between what we threw and a pattern. Possibly because we talk about dice throwing being luck/chance/random.

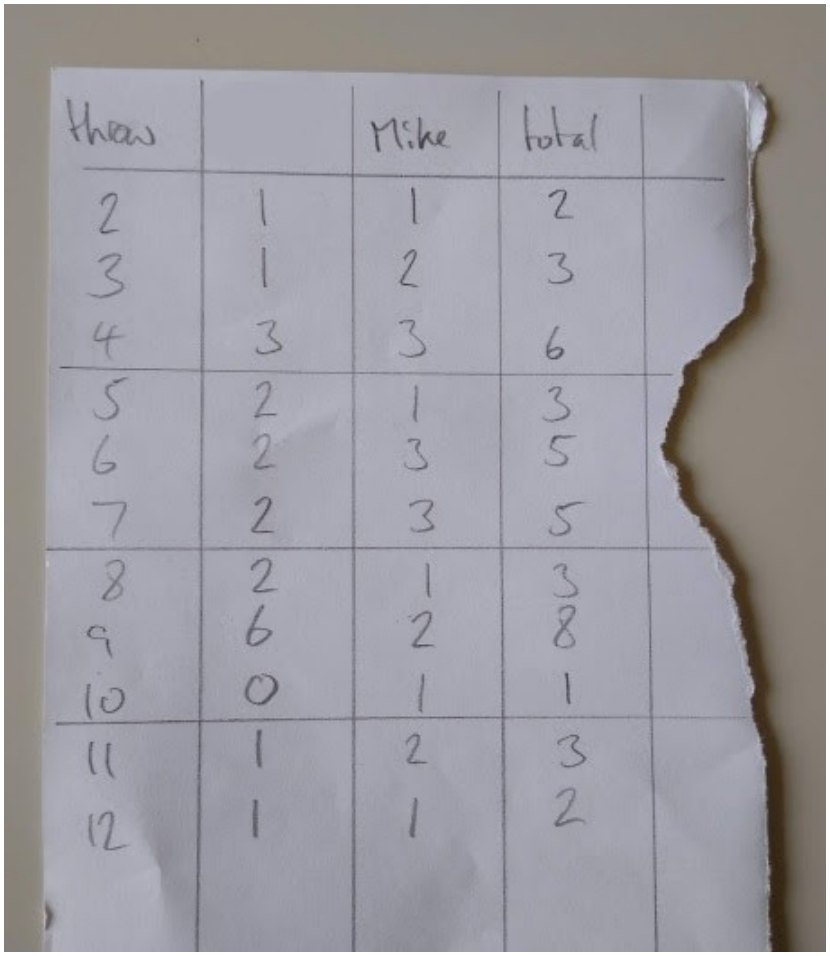
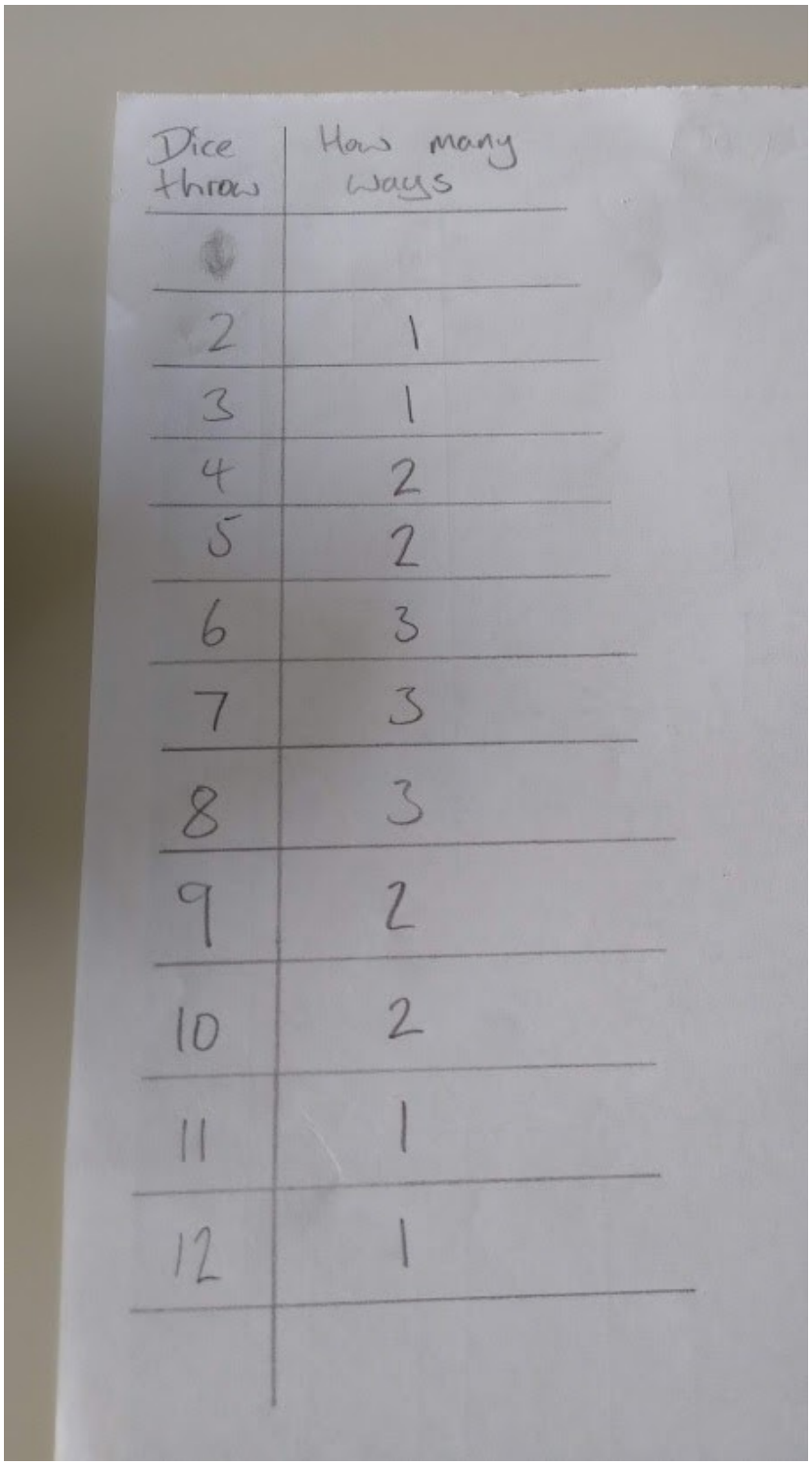


Table of dice rolls.

Next we converted these to a bar graph. Having previously made graphs on a computer he thought drawing it out was tedious! However, he enjoying colouring the bars in – to a point. Guess where.

Bar chart of dice rolls.

Finally we tallied up the ‘number friends’ (school term to describe two numbers which add up to a given number, e.g. 6 and 4 are number friends for 10) for each possible dice total. I took an quick decision that explaining a roll of 2 and 3 wasn’t the same as a roll of 3 and 2 was a leap too far – so we ignored those. It was at this point that he decided next time we played he would put all his balloons on middle numbers.



Dice throw	How many ways
2	1
3	1
4	2
5	2
6	3
7	3
8	3
9	2
10	2
11	1
12	1

Count of number friends for each dice total

However… I thought it would be neat to show him how to solve the problem with a computer. So I bashed out a quick function in R to simulate rolling a die. The code below samples numbers from 1 to 6, allowing replacement. The default rolls of the die are 10, but this can be changed by the user.

```
roll_die = function(n = 10){
  sample(1:6, size = n, replace = T)
}
```

Next we can roll our two dice a million times (6 yr old jaw on the floor) and sum the two numbers.

```
x = tibble(die_1 = roll_die(1000000),
       die_2 = roll_die(1000000)) %>%
  mutate(total = die_1 + die_2)
```

We can take the output from this to see how many rolls summed to each number, and because we know the

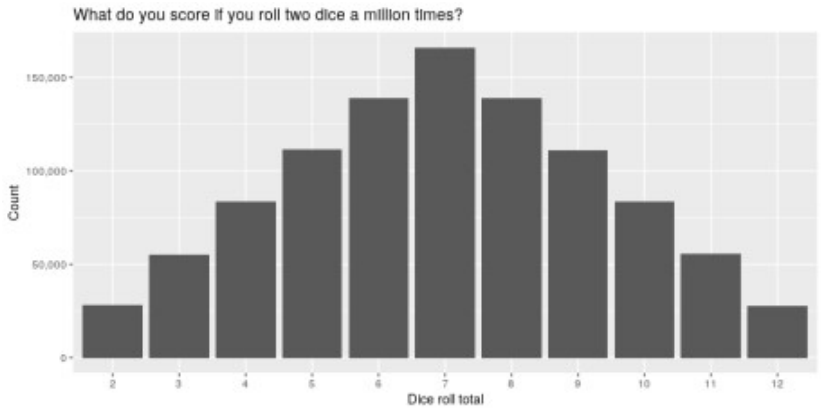
total number of rolls what the probability of scoring each total was (the latter omitted for 6 yr old).

```
x %>%
  count(total) %>%
  rename(roll_total = total) %>%
  mutate(occurence = scales::comma(n),
         probability = n / nrow(x)) %>%
  select(-n)
```

roll_total	occurrence	probability
2	27,953	0.0280
3	55,203	0.0552
4	83,396	0.0834
5	111,553	0.112
6	138,922	0.139
7	166,150	0.166
8	138,913	0.139
9	111,012	0.111
10	83,533	0.0835
11	55,703	0.0557
12	27,662	0.0277

We can also graph the results from the above table:

```
x %>%
  ggplot(aes(as.factor(total))) +
  geom_bar() +
  scale_y_continuous(labels = scales::comma) +
  labs(title = "What do you score if you roll two dice a million times?",
       x = "Dice roll total",
       y = "Count")
```



Finally, we print out all the distinct dice combinations for each total. I'll not print the result here, but the code is below. We can also tally the number of distinct combinations (table below).

```
# List combinations
lapply(2:12, function(i){
  x %>%
    filter(total == i) %>%
    distinct()
})

# Count combinations
y = lapply(2:12, function(i){
```

```
  y = x %>%
    filter(total == i) %>%
    distinct()

  tibble(total = i,
         combinations = nrow(y))
})
do.call("rbind.data.frame", y)
```

Roll total	Combinations
2	1
3	2
4	3
5	4
6	5
7	6
8	5
9	4
10	3
11	2
12	1

It’s a fun little exercise and there are many layers of learning for children of different ages.