Why is binomial expansion so special?

Binomial Expansion is symmetric, it is the foundation of probability, and it is a new way to look at algebra. I'll demonstrate each one by one, below:
What is $(a+b)^3$?

School taught me this:
$(a+b)(a+b)(a+b)$

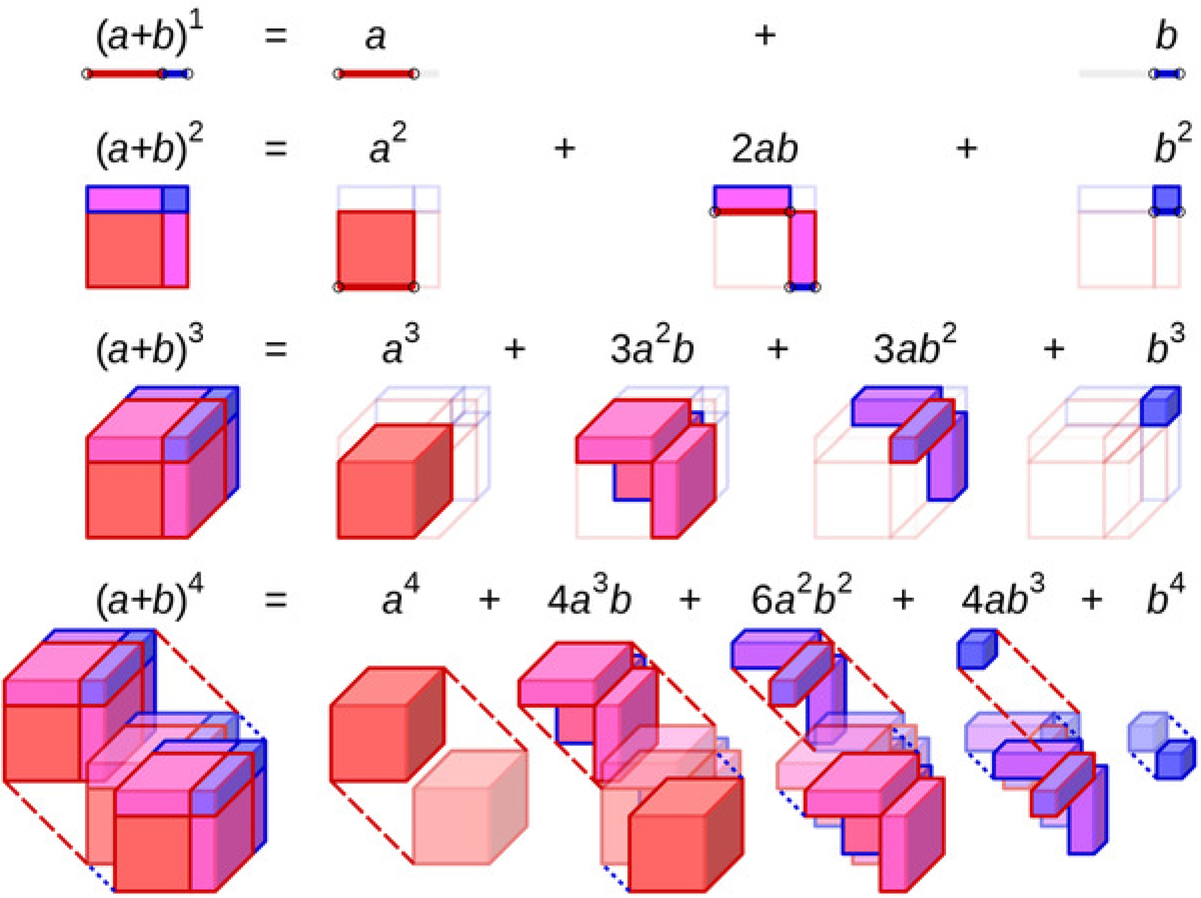
Then:
$a(a+b)(a+b)+b(a+b)(a+b)$

Then:
$a(a^2+2ab+b^2)+b(a^2+2ab+b^2)$

Then:
$a^3+2a^2b+ab^2+a2b+2ab^2+b^3$

Combining similar things:
$a^3+3a^2b+3ab^2+b^3$

What is the right way?
The right way is to ask, what does it mean? What is the pattern here?
Not only we need to see the pattern, but also one step before and after it:
Here is the pattern: which is the "why" part.



Here is how to think about the pattern, the "what is going on" part:
For this part, I am going to use Julia to demonstrate the link to the Binomial Triangle (aka Pascal's Triangle, even though it was discovered several times before Pascal).

If you are new to programming, just look at the results:

```
# importing a special package in Julia
using DSP
```

Now we can use convolution (I’ll explain):

```
# define a baze to start from
baze = [1];
# define a kernel that is to be convoluted
kernel = [1 1];
# repeat one time:
baze = conv(baze, kernel)
# result
>  1  1
```

Note that an ordered list of numbers is called a vector. In the above example, our kernel is a vector, but so is the base.

```
# vector a
a = [4 9]
# vector b
b = [8 5]
# dot product a.b:
```

$a \cdot b = ax \times bx + ay \times by$

Multiply x elements of a and b, then y elements of a and b, then add them all up. In Julia:

```
# define two vectors at the same time
a , b = [4 9] , [ 5 8]
# simple one verb command to get their dot product:
dot(a,b)
# output
> 92
```

Feel free to double-check this: 4 * 5 + 9 * 8 = 92
Now, convolution is like a moving dot product:

```
1 1
    1 1
-------
  0 1 0  = 1

  1 1
  1 1
------
  1 1 = 2

  1 1
1 1
------
0 1 0 = 1
```

Returning to our convolution example in Julia, the first convo is:

```
using DSP
baze = [1];
kernel = [1 1];

baze = conv(baze, kernel)

> 1 1

# note: we write baze so that it does not conflict
# with built-in Julia keyword: base.
```

Note the pattern in the second and third and fourth:

```
using DSP
baze = [1];
kernel = [1 1];


base1 = conv(baze, kernel)
base2 = conv(base1, kernel)
base3 = conv(base2, kernel)
base4 = conv(base3, kernel)
base5 = conv(base4, kernel)
base6 = conv(base5, kernel)
base7 = conv(base6, kernel)

println(base1, "\n",
          base2, "\n",
          base3, "\n",
          base4,  "\n",
          base5, "\n",
          base6, "\n",
          base7)

# resulting in:

        [1 1]
       [1 2 1]
      [1 3 3 1]
     [1 4 6 4 1]
    [1 5 10 10 5 1]
   [1 6 15 20 15 6 1]
 [1 7 21 35 35 21 7 1]
```

Cool! Now look at the third row: the pattern is 1 3 3 1. Is it relevant to:
$a^3+3a^2b+3ab^2+b^3$?

Look at the coefficients of each term:
a^3
coefficient is 1. (a * 1 = a)

$3a^2b$
coefficient is 3. (a^2 * 3 = 3a^2)

$3ab^2$
coefficient is 3. (a * 3 = 3a)

b^3
coefficient is 1. (b * 1 = b)

We established a pattern, so we are halfway through predicting this for any $(a+b)^n$.
But don't just stop there. The Binomial Triangle can also be found as powers of 11. :)

```
row1, row2, row3, row4 = 11^1, 11^2, 11^3, 11^4
println(row1,"\n",
          row2,"\n",
          row3,"\n" ,
          row4  )
# result:
11
121
1331
14641
```

Neat! But why do we need convolution if we can do this so simply? You'll see later why.
We have come a long way from a mere:
$(a+b)^3$

Haven't we? **This is how mathematics should be taught—everything is inter-related.** But frequently they're teaching everything in isolation—which means they don't really know it.

So now pay attention to the powers/exponents:
a is to the power of 3 in the first term, while b is to the power of 0:
$a^3*b^0=a^3*1=a^3$

a is to the power of 2 in the second term:
$a^2*b^1=a^2*b$

b's exponent is rising by 1, as a's is falling by 1:
$a^1*b^2=a*b^2$

and finally:
$a0*b3=1*b3=b3$

We have a new pattern, also called the combinatorics pattern:
$\binom{3}{0}, \binom{3}{1}, \binom{3}{2}, \dots, \binom{3}{3}$

Each tall bracket can be written as nCk
, or n choose k. So the first one can be re-writen as $3C0$
to mean that we have 3 cards; how many ways exist there to draw exactly 0 different cards?
Newton discovered a neat formula for it:

$$C_k(n) = \binom{n}{k} = \frac{n!}{k!(n-k)!}$$
$$C_0(3) = \binom{3}{0} = \frac{3!}{0!(3-0)!} = \frac{1}{1} = 1$$

By this convention, there is only one way to choose 0 items from n items.
Newton's formula is based on the idea of a factorial (another inter-link to our previous topics).

A factorial is simply a sum of consecutive products:

```
#   n! if n= 3

3 * 2 * 1
> 6

# short-cut in Julia:

factorial(3)
> 6
```

What about 2 items out of 3? How many different ways of choosing this is possible?
The slow method:

```
n = 3
k = 2

factorial(3)/ (factorial(2)* (factorial(3-2)))

> 3
```

Three exact ways to choose 2 items out of 3. Let's see it in practice and what it exactly means:
You have a choice of 3 ice-cream flavours. Out of 3, how many unique combinations are possible when you choose only 2 ice-creams each time?

```
using Combinatorics
things = ["chocolate" "vanilla" "raspberry"]
collect(combinations(["chocolate" "vanilla" "raspberry"], 2))

# result:

 ["chocolate", "vanilla"]
 ["chocolate", "raspberry"]
 ["vanilla", "raspberry"]
```

Getting interesting right? Who would've thought that boring-looking
$(a+b)^3$

has such an interesting connection?

The connection is this:
for 3C2, we have 3 combinations possible. Hence for nCk we have q combinations:
When a is the power of 3, and b is to the power of 2—aka 3C2—the result is 3, which is the coefficient for the 2nd term in the expansion.

Let's see coefficients computed for each of the terms in: $a^3+3a^2b+3ab^2+b^3$
0th term is where there is no b: a^3

```
n = 3
k = 0

factorial(n)/ (factorial(k)* (factorial(n-k)))
> 1
```

So 1 is the coefficient for a^3
1st term is decided by the fact that b^1
What is the coefficient for 1st term when our highest power is 3? $(a+b)^3$

```
n = 3
k = 1

factorial(n)/ (factorial(k)* (factorial(n-k)))

> 3
```

Niice! It matches the coefficient here : $3a^2b$

```
n = 3
k = 2

factorial(n)/ (factorial(k)* (factorial(n-k)))

> 3
```

Matching: $3ab^2$

```
n = 3
k = 3

factorial(n)/ (factorial(k)* (factorial(n-k)))

> 1
```

Matching: b^3

Given all this inter-linked ideas, we can predict and solve a binomial expansion under any reasonable n, by using the heuristics developed above.
Solve: $(t+u)^5$

Well, from powers of 11 we know:

```
  11^5
> 161051
```

The coefficients are:
1 then 6 then 10 then ...wait what? Why did I say what? because the Binomial Triangle is symmetric. Whatever a can have, so can b. Here, we expected the next number to be 6, but we don't have symmetricity. Perhaps using 11^n is not such a good idea.
Back to convolution:

```
base5
[1 5 10 10 5 1]
```

There!
So the first coefficient is 1,
then 5
then 10
then 10
then 5
then 1

Giving us a skeleton:
$a+5ab+10ab+10ba+5ba+b$

Let's add the powers. We know a starts high and becomes 0, and b vice versa:
$a^5+5a^4b+10a^3b^2+10b^3a^2+5b^4a+b^5$

Let's test it in Julia:

```
# import SymPy library from Python
using SymPy

#define symbolic variables
@vars a b

#solve
expand((a+b)^5)
```

Result: $a^5+5a^4b+10a^3b^2+10b^3a^2+5b^4a+b^5$

Okay so what happens if you don’t have Julia to compute for you?
You see the pattern in the Binomial Triangle:

```
1.         [1 1]
2.        [1 2 1]
3.       [1 3 3 1]
4.      [1 4 6 4 1]
5.     [1 5 10 10 5 1]
6.    [1 6 15 20 15 6 1]
7.  [1 7 21 35 35 21 7 1]
```

Look at the 4th row. How did we get the 5 in the 5th row? By adding 1+4 in the 4th row! 10? by adding the left 4 to central 6 and another 10 by adding central 6 to the right 4. (On both sides, the triangle is flanked by 1 btw).
Notice that each row tells us what the next row is, just like a linked list !

This is how mathematics should be taught. Understand and you don’t need to memorize. Enjoy it. Mathematics is beautiful!