VIM keybindings/tips and tricks that I use

Vim has this idea that the vim's interface is in itself a programming language, that being said, different key combinations have different effects and once you combine all these combinations you can create new key effects, similar to what we do in a programming language.

1. Enter the vim editor by
   >> vim hello.txt

2. Now you are in the normal mode, you can change to insert mode where you can make edits to the opened file by pressing
   >> i

3. So pressing i changes from normal mode to insert mode and now you can save your work by entering
   >> :w

4. To quit the editor
   >> :q

5. To save your work and quit simultaneously you can
   >> :wq

6. To split the vim editor into two parallel editors wherein the changes made in one split window will simultaneously show up on the other windows as well you can
   :sp

7. When you are in the **normal mode** you can use the keys on the alphabet pad on your keyboard to move up/down/left/right so as to save time moving your finger all the way to the end of keyboard reaching out to the arrow keys LOL
   >> h -> left
   >> l -> right
   >> k -> up
   >> j -> down

8. Now since moving letter by letter will kinda slow you down again **(normal mode)**, you can use the following combination to move word by word
   >> w (to move from word to word)
   >> b (to move back word to word)
   >> $ (to move to the end letter of each word)

>> 0 (to move to the starting letter of each word)

9. Talking about moving fast (**normal mode**) here and there we need key combinations to page-down and page-up too
   >> ctrl + u (to go up the file)
   >> ctrl + d (to go down the file)

10. There is also a way to move the entire buffer of the file (**normal mode**)
    >> G move all the way up
    >> gg moves all the way down

11. Once you are in the normal mode, **press v to enter the VISUAL** mode. Here you can use the normal navigation HJKL keys to copy a chunk of text and then **press y to copy** and **press p to paste** the data still being in the normal mode!

12. Now since we cans elect all the text back and forth we can use the " ~ " tilda character to flip the alphabet casing. I.e Hello World will become hELLO wORLD.

13. Since we have already established vim's interface is a programming language we can move up or down the lines in **visual mode by** providing the number of lines we wish to move up or down and then providing the key that moves up or down. I meant
    7j will take you 7 lines below.
    20k will take you 20 lines upwards.

14. One of my favourite effects that vim offers is the repetition of the previous edit made in the file or code.

    Let's say we have

```
Int main(void)
{
int b=20 , a=10, c=30;
int a = 10;
printf("%d %d", a);
printf("%d %d", b);
return 0;
}
```

    Now I need to append c in both the printf lines since I have 2 %d's and I needed to print c in both the print statements (pls note I'm just trying to give a simple example, however you'd like to scale this feature is totally up to you)

    Instead of writing c both the times in the print statements you can use the . (full stop) to

repeat the changes you did before in insert mode, Note . can only be used in normal mode to replicate the changes you made in the insert mode.
What I meant to say is if I write this

```
Int main(void)
{
int b=20 , a=10, c=30;
int a = 10;
printf(“%d %d”, a, c);
printf(“%d %d”, b);
return 0;
}
```

Now after taking my cursor next to b in the next line, I can simply press . and the c will autofill itself. This is not limited to characters but with whole lines as well.

This list will keep updating with more useful tips when I come around them. Thanks