Java regex notes

1. **[A-Z]** - Any character between A-Z
2. **[^A-G]** - Any character other than A-G
3. **\\s** - Search for whitespace
4. **\\S** - Search for not whitespace
5. **\\w{2,20}** - Match any character with min 2 and 20 characters
6. **[A-Za-z]{2,20}** - Match any character with min 2 and 20 characters
7. **\\d** - Any digit
8. **\\D** - Not a digit
9. **{5}** - Occured 5 times
10. **A[KART]** | **C[SAD]** - Start with A or C and after that any of the characters specified in [] should occur
11. **{n,}** - Min n characters
12. **{,m}** - Max n characters
13. **(\\{{1,})** - search for one or more '{'. (**Also (\\{+)).**
14. Whenever we want to search for any **., ^, *, {, }, [,], \, |, (,), +,?** We have to use **double backslash (\\)**.
15. **+** is equivalent to **{1,}.**
16. **.** - It matches anything
17. **\\w** is equivalent to **[A-Za-z0-9].**
18. **\\W** is equivalent to not **\\w.**
19. * - anything that occurs **0 or more times.**
20. **? -** That doesn't need to exist (Similar to **or nothing**)

21. ***b*** - matches the boundary between a word and a non-word character. It's most useful in capturing entire words (for example by using the pattern *\w+\b*).

Extra note :
One concept that we will not explore in great detail in these lessons is *back referencing*, mostly because it varies depending on the implementation. However, many systems allow you to reference your captured groups by using *\0* (usually the full matched text), *\1* (group 1), *\2* (group 2), etc. This is useful for example when you are in a text editor and doing a search and replace using regular expressions to swap two numbers, you can search for "(\d+)-(\d+)" and replace it with "\2-\1" to put the second captured number first, and the first captured number second for example.