learnings learnt

IFS=$'\n' read -d "" -a arraylines < file.txt

- IFS → Input Field separator
- It reads lines (all ~~words~~ before \n is read as one element).
- read → this is the command to help us read something basically
- -d → It refers to the delimiter and what follows after that is basically what you would have as your delimiter.
  In our case, the next part shows an empty string which highlights that the delimiter is set to none
- -a → appending our lines inside our "arraylines" bash array
- < file.txt → the input that is being fed to our system is coming strictly from our file.txt basically.

note if-else statement

```
if ["$TARGET" == 0]; then
    // have your statements here
done
```

to increment:

```
$TARGET += 1   X
TARGET=$((TARGET + 1))
```

→ To avoid word splitting, make sure you do a double quotes around the name of your array basically.

```
for line in "${arraylines[@]}"; do
    // write your statements
done
```

Logic:

We will iterate through the bash array after putting all our lines in here. We will have our target variable as 9 which is our destination line that we want to be able to get to. Our count variable will be incrementing continuously - if our count reaches the value of our target variable then we want to echo that line.

Out of that for loop, we will echo an empty line or "" in case there was no empty line available for usage.