2.2 Coding

- The Ocaml code is shown below:

```
let s = read_line();;

let list = List.map int_of_string (Str.split (Str.regexp ", ") s);;

let rec quicksort = function
  | [] -> []
  | head::tail -> let smaller, larger = List.partition ((>) head) tail in
quicksort smaller @ (head::quicksort larger)
;;

let ll = quicksort list;;

List.iter (Printf.printf "%d ") ll;;
```

  Firstly, take the standard input as a string and save it to **s**. Then use `Str.split` and `Str.regexp` to split `s` into multiple strings of integers, and use `List.map` with `int_of_string` to transform into the list of integers.

  After that, the function `quicksort` is defined as a recursive function. If the input is empty list, it returns an empty list. If not, the list will be parsed into two lists using `List.partition`, one contains all the integers smaller or equal to the pivot `head`, the other with all the larger integers. Then, quicksort the two split lists until empty.

  Finally, assign the returned list of `quicksort` to `ll` and use `List.iter` to print all the integers in the sorted list.

- The average time complexity is $\mathcal{O}(n \log n)$.