Chapter – 3 Functions Summary :-

Function is the type of procedure or routine that performs the specific task.

Rules to write Functions are :-

Small

- The first rule of functions is that they should be small.
- The second rule of functions is that they should be smaller than that.

Blocks and Indenting

- This implies that the blocks within if statements, else statements, while statements, and so on should be one line long.

Do One Thing

- FUNCTIONS SHOULD DO ONE THING . T HEY SHOULD DO IT WELL.
- THEY SHOULD DO IT ONLY .

One Level of Abstraction per Function

- We need to make sure that the statements within our function are all at the same level of abstraction.
- Mixing levels of abstraction within a function is always confusing.

Reading Code from Top to Bottom: The Stepdown Rule

Switch Statements

- Switch statements can be tolerated if they appear only once, are used to create polymorphic objects, and are hidden behind an inheritance relationship so that the rest of the system can’t see them.

Use Descriptive Names

- The smaller and more focused a function is, the easier it is to choose a descriptive name.
- A long descriptive name is better than a short enigmatic name.
- A long descriptive name is better than a long descriptive comment.

Function Arguments

- The ideal number of arguments for a function is zero (niladic).
- Next comes one (monadic), followed closely by two (dyadic).
- Three arguments (triadic) should be avoided possibly.

- Output arguments are harder to understand than input arguments.
- We don’t usually expect information to be going out through the arguments.

Avoid the following :-

-Common Monadic Forms except like
void loginFailedNumberOfTimes(int numberOfTimes).

-Flag Arguments
-Dyadic Functions
-Triads
-Argument Objects

Verbs and Keywords :-

- In the case of a monad, the function and argument should form a very nice verb/noun pair.

Have No Side Effects :-

- The function should do only one thing that is promised by its name and does not effect other parts of the progra.

Output arguments should be avoided.

Command Query Separation:

- Functions should either do something or answer something, but not both.
- Either your function should change the state of an object, or it should return some information about object.
- Doing both often leads to confusion.

-Prefer Exceptions to Returning Error Codes
-Extract try/ catch blocks.
-Error Handling Is One Thing.