Common Rules.

1. **Use Intention-Revealing Names:** The name of a variable, function or class, should tell why it exists? What is does? How it is used? If a name needs a comment, then it does not reveal its intent. int d;(bad) int numberOfDays;(good)
2. **Avoid Disinformation:** Programmer must avoid leaving false clues that obscure the meaning of code. For example, don’t refer a group of accounts as accountList unless it is actually a List. The word List means something specific to programmers. If the container holding accounts not a List then accountGroup, branchOfAccounts or just accounts.
3. **Make Meaningful Distinctions:** Sometimes programmer writes the code just to satisfy the compiler. Like, two variables doing same work in a single block. So, programmer called date1, date2. Surly there are distinct reasons to use two variables. Programmer should identify that reason and declare the variable names accordingly. Like, startDate-endDate, joiningDate-lastDate, sourceDate-targetDate etc.
4. **Use Pronounceable Names:** uA large portion of human brain has evolved to deal with spoken language. Programmer should take advantage of this. Make your name pronounceable. Like, Date modymdhms;(Bad). Give the name Date modificationTimeStamp;(Good)
5. **Use Searchable Names:** Single letter names and numeric constants have a particular problem in that they are not easy to locate across a body of code. Like, MAXIMUM\_STUDENT\_IN\_A\_CLASS easy to grep, but number 9 could be difficult to identify.
6. **Avoid Encoding:** It is unnecessary mental burden when someone reading code any trying to understand the logic. String phoneNumberString;(Bad). String phoneNumber;(Good).
7. **Avoid Mental Mapping:** For professional programmer clarity is king. So, no mental mapping. Like, host name is h, and port number is pb.
8. **Class Names:** Classes and objects should have noun or noun phrase. Like, Customer, CustomerAccount, EmployeeAddress. Class and object should not be a verb.
9. **Method Names:** uMethod should have verb or verb phrase names like deletePage(), saveEmployee(), createEmployee(), openDBConnection() etc.
10. **Pick One Word Per Concept:** Pick one word for one abstract concept and stick with it. Like, fetch, retrieve, get all equivalent meaning. So, if get already used in existing code then programmer should stick with it. Similarly, Handler, Controller, Driver. It should not be DeviceDriver and ProtocalController.
11. **Don’t Pun:** Using the same term for two different ideas is a pun. Like, method names are add(), and they are doing add data into list, add two integers, add two strings. But the semantics are different. So, use of concat or append or insert will provide more clear behavior about those methods.
12. **Use Solution Domain Names:** Remember the programmer who going to read your code will be programmers. So, go ahead and use the computer science, algorithm, math terms, design pattern names. Like, JobQueue, ConnectionFactory. When any programmer reads JobQueue, he/she knows what is it?
13. **Use Problem Domain Names:** Use problem domain name help to understand the purpose of the problem the piece of code going to solve. Like, applyRaid0. Here Raid0 is domain specific it data center. So, when any programmer reads this method he/she knows what is Raid0 if he/she is part of the data center domain.
14. **Add Meaningful Context:** Sometime the situation occurred when programmer has to think more in depth and choose the meaning full name. Like, Address is a class containing persons address. But in the same program address is appearing for permanent address, home address, office address. In this situation with that phrase address should be declared.
15. **Don’t Add Gratuitous Context:** Imagine your application is for Bangalore Travel Agency. Many programmers add BTA for variable, method, class. Like, BTAEmpolyee, BTACustomer. It should be only Employee, Customer.

Functions.

1. Small:
2. Blocks and Indenting:
3. Do One Thing:
4. One Level of Abstraction per Function:
5. Use Descriptive Name:
6. Function Arguments:
7. Common Monadic Forms:
8. Flag Arguments:
9. Dyadic Function:
10. Triads:
11. Argument Objects:
12. Argument Lists:
13. Verbs and Keywords:
14. Output Arguments:
15. Command Query Separation:
16. Prefer Exceptions to Returning Error Codes:
17. Extract Try/Catch Blocks:

Comments.

1. Inappropriate Information:

2. Obsolete Comment.

3. Redundant Comment.

4. Poorly Written Comment:

Variable Names.

Java.

1. Avoid Long Import Lists by Using Wildcards:
2. Don’t Inherit Constants: