PROGMASTERS

reboot your life!

LEVELS OF DEVELOPERS

	Level 0	Level 1	Level 2	Level 3	PROG-masters level
Computer Science					
data structures	Doesn't know the difference between Array and LinkedList	Able to explain and use Arrays, LinkedLists, Dictionaries etc in practical programming tasks	Knows space and time tradeoffs of the basic data structures, Arrays vs LinkedLists, Able to explain how hashtables can be implemented and can handle collisions, Priority queues and ways to implement them etc.	Knowledge of advanced data structures like B-trees, binomial and fibonacci heaps, AVL/Red Black trees, Splay Trees, Skip Lists, tries etc.	1
algorithms	Unable to find the average of numbers in an array (It's hard to believe but I've interviewed such candidates)	Basic sorting, searching and data structure traversal and retrieval algorithms	Tree, Graph, simple greedy and divide and conquer algorithms, is able to understand the relevance of the levels of this matrix.	Able to recognize and code dynamic programming solutions, good knowledge of graph algorithms, good knowledge of numerical computation algorithms, able to identify NP problems etc.	1,5
Software Engineering					
version Control System [1]	Doesn't know about version control system so does not use it. Files are not tracked in version control.	Basic use of version control system. All team members push to the main branch.	Good use of VCS and it's features. Team members do feature/task level branching with pull requests being a commonplace.	Advanced use of VCS and leverages most of it's features in addition to branching and merging. For example git bisect if using git.	1,5
automated testing	Does not write any form of automated tests. Relies fully on manual testing if any testing is done.	Has written automated unit tests and comes up with good unit test cases for the code that is being written	Has unit and/or integration tests in place. May have functional tests too. Has written code in TDD manner	Understands and is able to setup automated functional, load/performance and UI tests. Tests are like a first line of defense for catching bugs and tracing requirements.	2
database	Thinks that Excel is a database	Knows basic database concepts, normalization, ACID, transactions and can write simple selects	Able to design good and normalized database schemas keeping in mind the queries that'll have to be run, proficient in use of views, stored procedures, triggers and user defined types. Knows difference between clustered and non-clustered indexes. Proficient in use of ORM tools.	Can do basic database administration, performance optimization, index optimization, write advanced select queries, able to replace cursor usage with relational sql, understands how data is stored internally, understands how indexes are stored internally, understands how databases can be mirrored, replicated etc. Understands how the two phase commit works.	1
Programming					
problem decomposition	Only straight line code with copy paste for reuse	Able to break up problem into multiple functions	Able to come up with reusable functions/objects that solve the overall problem	Use of appropriate data structures and algorithms and comes up with generic/object-oriented code that encapsulate aspects of the problem that are subject to change.	2,5
code organization within a file	no evidence of organization within a file	Methods are grouped logically or by accessibility	Code is grouped into sections and well formed	File has license header, summary, well commented, consistent white space usage. The file should look beautiful.	2
code readability	Mono-syllable names	Good names for files, variables classes, methods etc.	No long functions, comments explaining unusual code, bug fixes, code assumptions	Code assumptions are verified using asserts, code flows naturally – no deep nesting of conditionals or methods	2
error handling	Only codes the happy case	Basic error handling around code that can throw exceptions/generate errors	Ensures that error/exceptions leave program in good state, resources, connections and memory is all cleaned up properly	Codes to detect possible exception before, maintain consistent exception handling strategy in all layers of code, come up with guidelines on exception handling for entire system.	1,5
refactoring and rewrite	Never allocates times for any refactoring and/or rewrite. Just make it work mentality prevails in the team.	Manages time for some code refactoring and has a mentality to leave the code better than it was. Does not allocate time for rewrite.	Refactoring and rewrite are priorities. Allocates and uses time to refactor low performing and badly written parts. Manages time to rewrite parts which are old and causing problems.	Refactoring and rewrite is ingrained in the process. Doing a rewrite of old components/parts is taken as an opportunity.	1,5
Security					
App level [2]	Is reluctant to take any measures against app level security.	Is aware of application level security vulnerability still fixes them slowly as it comes to priority late	As and when informed about any application level security vulnerability fixes the issues in an acceptable amount of time.	Actively looks for any application level vulnerability in the code and makes fixing security vulnerability a high priority fixing it very fast.	1,5
Tools					
project management software [3]	Does not use any project management software.	Uses project management software but partially and with less planning.	Takes advantage of a good project management software with roadmap, plans and other artifacts in place.	Takes the most advantage out of the project management software uses it to track progress and extract useful reports out of the software.	2
IDE	Mostly uses IDE for text editing	Knows their way around the interface, able to effectively use the IDE using menus.	Knows keyboard shortcuts for most used operations.	Has written custom macros	2
Soft Skills					
communication [4]	Cannot express thoughts/ideas to peers. Poor spelling and grammar.	Peers can understand what is being said. Good spelling and grammar.	Is able to effectively communicate with peers	Able to understand and communicate thoughts/design/ideas/specs in a unambiguous manner and adjusts communication as per the context	2

[1] Git is the most popular distributed VCS. This section also covers the use of VCS management software like Github. etc. | [2] Application level vulnerability like SQL injection, CSRF, cross site 1scripting or even wrong ACL, misconfigured settings etc. | [3] Jira is one of the popular project management software with good reporting capabilities. | [4] This is an often under rated but very critical criteria for judging a programmer. With the increase in outsourcing of programming tasks to places where English is not the native tongue this issue has become more prominent. I know of several projects that failed because the programmers could not understand what the intent of the communication was.