Note:

1. The suggested plan is based on information received from the students who have received internships. You may go through the entire schedule and adjust it according to your strong and weak areas.
2. While we have tried to cover all the important topics, you are still requested to refer your lecture notes, textbooks and handouts for a thorough understanding of the topics. The basics of all the second-year courses should be clear.
3. Technical Interviews regularly include problems to be solved during the interview. Hence, prepare yourself by solving different problems and interrupting yourself by asking questions while solving as interviewers will do that.
4. A good preparation strategy would be to finish the sections as per schedule. Additionally, having a list of topics you are weak in would be a useful reference for the night just before the Interviews.
5. In case of doubts, do reach out to the seniors who have secured Internships.
6. If you find any error in this document, do let us know at placement@goa.bits-pilani.ac.in.

Growth is an uncomfortable journey from an uncomfortable position to a more comfortable position.

Introduction:

IT companies primarily ask questions based on DSA in the coding rounds and interviews. Sometimes questions from OOP and DBMS are also asked. OS, Networks and Design also form part of some companies.
Hence,

1. **Primary Priority (60%):** DSA
2. **Secondary Priority (25%):** OOP, DBMS
3. **Tertiary Priority (15%):** OS, Networks and Design

For students in their initial stage of preparation or those who are from a non-CS background, but aiming for IT companies, *Cracking the Coding Interview (CTCI)* book is highly recommended because it contains necessary theory of each topic given in a nutshell, and also a collection of relatively easy and medium questions.

InterviewBit and GeeksForGeeks have company-specific and real interview questions, and often times highly likely to be asked as well.

Confusion of Programming Language:

It is generally recommended to use C++ over C because C++ has an STL i.e. standard library, which has inbuilt functionalities, which C doesn't have. Additionally, they have a similar syntax, making it easier to shift for one from C to C++. C++ has inbuilt support for variable sized arrays (*std::vector*), self-balancing red-black trees (similar to AVL) (*std::set, std::map*), hashmap (*std::unordered_map*) etc. Hence, they come really handy while solving coding tests and interview questions, which generally have a time constraint, and implementing these on your own would be too long.

Alternatively, if you feel that you have a strong preference for Java, it also has an almost similar standard library but is generally slower. Hence, C++ is recommended over Java for performance, when an optimal solution can exceed the time limit in Java while passing in C++.

Python is even slower than Java and highly discouraged for coding tests.

As a rule of thumb, we can say that Java takes 2x running time of C++ code, and Python takes around 10x of C++.

References:

- Cracking the coding interview (CTCI) *(Preferred)*
- InterviewBit *(Preferred)*
- Geeksforgeeks *(Preferred)*
- hackerrank.com
- leetcode.com
- https://www.codechef.com/certification/data-structures-and-algorithms/prepare#foundation
- Introduction to Algorithms – CLRS
- https://cp-algorithms.com/

CAUTION!
There are a lot of resources available online for IT Preparations. Students often keep jumping from one website to another and finally end up reaching nowhere. Hence, stick to fixed references and understand them properly.

placement@goa.bits-pilani.ac.in

Week 1

- Basic input output and Errors that you can get while submitting your code to an online judge. (https://www.hackerearth.com/practice/basic-programming/input-output/basics-of-input-output/tutorial/)
- Taking multiple inputs in a single line in Python (https://www.geeksforgeeks.org/input-multiple-values-user-one-line-python/)
- Fast input-output in C++ (https://www.geeksforgeeks.org/fast-io-for-competitive-programming/)
- C++ STL
    - https://www.hackerearth.com/practice/notes/standard-template-library/
    - https://www.topcoder.com/community/competitive-programming/tutorials/power-up-c-with-the-standard-template-library-part-1/
    - https://www.topcoder.com/community/competitive-programming/tutorials/power-up-c-with-the-standard-template-library-part-2/
- Time Complexity
    - Slides from DSA course or CLRS textbook
    - https://www.interviewbit.com/courses/programming/topics/time-complexity/
- Maths
    - InterviewBit (https://www.interviewbit.com/courses/programming/topics/math/)
    - CTCI: Chapter 6
    - Basic Probability
    - *Additional Topics:* Modular Math, Primality Testing, Sieve Algorithm, Modular Exponentiation, Euclidean GCD Algorithm
- Arrays, Sorting
    - InterviewBit (https://www.interviewbit.com/courses/programming/topics/arrays/)
    - CTCI
- Divide and Conquer algorithms (standard ones) (https://www.geeksforgeeks.org/divide-and-conquer/#standard)
- Important Questions
    - https://www.geeksforgeeks.org/write-a-c-program-to-calculate-powxn/
    - https://www.geeksforgeeks.org/counting-inversions/

Week 2

- Bit Manipulation
    - InterviewBit
    - CTCI
    - HackerEarth (https://www.hackerearth.com/practice/basic-programming/bit-manipulation/basics-of-bit-manipulation/tutorial/)
    - GeeksForGeeks (https://www.geeksforgeeks.org/sum-of-xor-of-all-subarrays/)

placement@goa.bits-pilani.ac.in

- Binary Search
  - InterviewBit
  - CTCI
  - Aggressive cows (https://www.spoj.com/problems/AGGRCOW/) and its variants
  - Article (https://medium.com/@lohitmarodia/searching-linear-binary-and-their-applications-55d7e9eaa552)
  - Standard Library Functions for C++/Java for Binary Search (http://www.cplusplus.com/reference/algorithm/lower_bound/)

- Square Root Decomposition
  - GeeksforGeeks (https://www.geeksforgeeks.org/sqrt-square-root-decomposition-technique-set-1-introduction/)
  - Important Question: Egg dropping puzzle with 2 eggs and 100 floors (Throwing first egg from 10, 20, ..., 90, 100, and then doing a linear search in-between.) https://stackoverflow.com/questions/3974077/throwing-cats-out-of-windows?rq=1

Week 3

- String Questions
  - InterviewBit
  - CTCI Book (chapter 1)
- String Hashing
- Rabin Karp String Matching
- Z and KMP algorithm
- Two Pointers
- Linked Lists

(Refer GFG, InterviewBit, CTCI for these topics)

Week 4

- Stacks and Queues
- Recursion and Backtracking
- Hashing
- Greedy Algorithms
- Heaps and Maps
  - Usually, Heap Implementation is not asked, while application of Heap and Maps is asked
  - https://www.hackerrank.com/challenges/minimum-average-waiting-time/problem
    https://www.hackerrank.com/challenges/find-the-running-median/problem
    https://www.hackerrank.com/challenges/jesse-and-cookies/problem
  - For using a Hash map, C++ has std::map and std::unordered_map
    For heap, it has std::set, which can help maintain minimum element in O(log N), and unordered_set for O(1)

Week 5

- Binary and N-ary Trees, BSTs (InterviewBit and CTCI)
  - Basic operations in a BST (insertion, deletion, search, in-order successor, predecessor, different traversals) - iteratively and recursively
  - Augmentation of the node (struct) in a BST/tree (i.e. doing first DFS, storing some extra information into nodes themselves, and then using it to solve the original problem) (https://www.hackerrank.com/challenges/is-binary-search-tree/problem)
  - DP on trees (https://codeforces.com/blog/entry/20935)

placement@goa.bits-pilani.ac.in

- Trie (https://www.hackerearth.com/practice/data-structures/advanced-data-structures/trie-keyword-tree/tutorial/)
- AVL Rotations
- Segment trees/Interval trees (like range minimum query with point updates, range-sum query etc.) (https://www.hackerearth.com/practice/data-structures/advanced-data-structures/segment-trees/tutorial/)

- Dynamic Programming
  - Knapsack Problem, Rod cutting problem, String based Problems
    https://people.cs.clemson.edu/~bcdean/dp_practice/
    http://prismoskills.appspot.com/lessons/Dynamic_Programming/Chapter_01_-_Introduction.jsp

Week 6

- OOP
  - Why OOP?
  - Access modifiers: public/private/protected
  - Abstraction
  - Encapsulation
  - Inheritance
  - Polymorphism
  - Exception handling
  - Overriding vs overloading
  - Constructor – destructor
  - this/self pointer
  - Multithreading and synchronization
  - Thread-safety and thread-safe containers/collections
  - Deadlock
  - Lambda expressions
  - Range based for-loops / for-each and iterators
  - Association Composition Aggregation
  - Multiple inheritance in diamond problem
  - Object serialization/pickling
  - SOLID principles of object-oriented design (https://en.wikipedia.org/wiki/SOLID)

- Programming Language
  - C
    - Floating point precision issues
    - Operator precedence and short-circuiting conditions
    - Keywords based questions (static, volatile, const, extern, register, etc)
    - Compilation stages
    - Static vs dynamic memory allocation
    - Preprocessor/macros
    - Difference between pointer and array
    - Null pointer
    - Dangling pointer
    - Memory leaks
    - Stages in a function call/ internal stack usage
    - Variable scope rules

- Typecasting issues
- Why global variables are bad?
- Multithreading using pthread (basics)
- Mutex (basics)
- Semaphores (basics)

Thoroughly prepare one of the following three languages:

- C++
  - Copy constructor and assignment operator (the rule of three)
  - Struct vs object
  - Operator overloading
  - Reference variables vs pointers
  - Keywords based questions (virtual, inline, friend, etc)
  - Static functions in a class, STL - pair, tuple, string, vector, set, map, unordered_map
  - Multimap
  - Stack
  - Queue, deque
  - Templates
  - New and delete vs malloc and free
  - Function pointers
  - Opaque pointer
  - Pass by value / by reference / by pointer
  - Final class
  - Abstract class
  - Namespaces
  - Object slicing
  - Iostreams
- Java
  - Pass by value vs by reference
  - == vs equals() method
  - Wrapper classes
  - Keywords based (final, finally, finalize, super, abstract, synchronized, instanceof)
  - Generic classes
  - Java references vs C/C++ pointers
  - Interfaces
  - Garbage collection
  - JVM internals
  - Static methods
  - String immutability and other features
  - Stringbuilder/stringbuffer
  - Static class
  - Object class
  - Stream class
  - Object Cloning
  - Comparator and Comparable interface
  - Nested and inner classes
  - Collections - arraylist, treemap, hashmap, treeset, hashset, Queue, Stack, Deque
  - Concurrent Collections
  - Biginteger class

        - Circular memory reference
        - Memory leaks

    - Python
        - Keywords (some more) (important ones: yield, with, import, pass, in, is, global, nonlocal, del)
        - Duck-typing idiom
        - Garbage collection
        - Data types (string, list, tuple, set, dictionary)
        - Big integer
        - Collection: deque
        - Heap
        - Namedtuple
        - Global vs local variables
        - Packing and unpacking
        - Operator overloading
        - == vs is operator
        - Iterator objects
        - Immutable strings
        - Shallow copy- deep copy
        - Range() and xrange()
        - Looping techniques
        - Else block after loop
        - Function decorators
        - Closures
        - Constructor and destructor
        - Class methods vs static methods
        - Class/static variables
        - Circular memory reference and memory leaks
        - Python tricks
        - Code optimizations
        - Filter/Map/Reduce functions
        - Competitive coding tips
        - Permutation and combinations

- Database systems (https://www.geeksforgeeks.org/dbms/)
    - SQL operations (Select, join etc)
    - Recursive SQL select queries
    - ER modelling
    - Relational model and schema
    - B/B+ trees implementation
    - Normalization
    - https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions/
    - https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions-set-2/
    - Basics of NoSQL databases, NoSQL vs SQL databases

Week 7

Graph algorithms

- BFS
- DFS
- Cycle detection

placement@goa.bits-pilani.ac.in

- Dijkstra
- Topological sort (https://www.hackerearth.com/practice/algorithms/graphs/topological-sort/tutorial/)
- Connectivity
- Bipartite
- Hamiltonian/Eulerian path/cycle
- Minimal spanning tree - Prim's and Kruskal's
- Union-find (Disjoint-set union) (https://www.hackerearth.com/practice/data-structures/disjoint-data-strutures/basics-of-disjoint-data-structures/tutorial/)

Design Pattern

- https://www.geeksforgeeks.org/design-patterns-set-1-introduction/
- https://www.geeksforgeeks.org/singleton-design-pattern/

Microservice

- https://www.geeksforgeeks.org/microservices-introduction/

Note:
Once you complete the entire plan, you are suggested to practice company specific questions from InterviewBit and GeeksforGeeks.
Also, reach out to different seniors to understand their interview experiences.

placement@goa.bits-pilani.ac.in