NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Affiliated to

DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY UTTAR PRADESH, LUCKNOW

Evaluation Scheme & Syllabus

For

B.Tech - Second Year-Lateral Entry (B.Sc.)

(Effective from the Session: 2021-22)

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Bridge Courses for Lateral Entry Students Admitted Through (B. Sc.)
B.Tech (CS, CSE(IOT), ECE, IT, CSE(AIML), ME, CSE, CSE(DS), CSE(AI), BT)
EVALUATION SCHEME
SEMESTER-III

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	ACSE0101Z	Problem Solving using Python	3	0	0	30	20	50		100		150*	0
2	ACSE0151Z	Problem Solving using Python Lab	0	0	2				25		25	50*	0
		GRAND TOTAL										200*	

All the students must clear the above mentioned subjects of the first year (Semester-I) Engineering Program along with the second year (Semester-III) subjects.

*All Bridge Courses are zero (0) credit courses.

*Total and obtained marks are not added in the Grand Total.

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Bridge Courses for Lateral Entry Students Admitted Through (B. Sc.)
B.Tech (CSBS)
EVALUATION SCHEME
SEMESTER-III

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	ACSBS0103Z	Fundamentals of Computer Science	3	0	0	30	20	50		100		150*	0
2	ACSBS0153Z	Fundamentals of Computer Science Lab	0	0	4				25		25	50*	0
		GRAND TOTAL										200*	

All the students must clear the above mentioned subjects of the first year (Semester-I) Engineering Program along with the second year (Semester-III) subjects.

*All Bridge Courses are zero (0) credit courses.

*Total and obtained marks are not added in the Grand Total.

Abbreviation Used: -

L: Lecture, T: Tutorial, P: Practical, CT: Class Test, TA: Teacher Assessment, PS: Practical Sessional, TE: Theory End Semester Exam., PE: Practical End Semester Exam.

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Bridge Courses for Lateral Entry Students Admitted Through (B. Sc.)
B.Tech (CSE, CS, IT, AIML, AI, DS)
EVALUATION SCHEME
SEMESTER-IV

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	ACSE0202Z	Problem Solving using Advanced Python	3	1	0	30	20	50		100		150*	0
2	ACSE0252Z	Problem Solving using Advanced Python Lab	0	0	2				25		25	50*	0
		GRAND TOTAL										200*	

All the students must clear the above mentioned subjects of the first year (Semester-II) Engineering Program along with the second year (Semester-IV) subjects.

.

*All Bridge Courses are zero (0) credit courses.

*Total and obtained marks are not added in the Grand Total.

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Bridge Courses for Lateral Entry Students Admitted Through (B. Sc.)
B.Tech (BT)
EVALUATION SCHEME
SEMESTER-IV

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	ABT0201Z	Introduction to Biotechnology	3	0	0	30	20	50		100		150*	0
2	ABT0251Z	Introduction to Biotechnology Lab	0	0	2				25		25	50*	0
		GRAND TOTAL										200*	

All the students must clear the above mentioned subjects of the first year (Semester-II) Engineering Program along with the second year (Semester-IV) subjects.

*All Bridge Courses are zero (0) credit courses.

*Total and obtained marks are not added in the Grand Total.

Abbreviation Used: -

L: Lecture, T: Tutorial, P: Practical, CT: Class Test, TA: Teacher Assessment, PS: Practical Sessional, TE: Theory End Semester Exam., PE: Practical End Semester Exam.

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Bridge Courses for Lateral Entry Students Admitted Through (B. Sc.)
B. Tech (ECE, ME, IOT)
EVALUATION SCHEME
SEMESTER-IV

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	ACSE0201Z	Programming for Problem Solving using C	3	0	0	30	20	50		100		150*	0
2	ACSE0251Z	Programming for Problem Solving using C Lab	0	0	2				25		25	50*	0
		GRAND TOTAL										200*	

All the students must clear the above mentioned subjects of the first year (Semester-II) Engineering Program along with the second year (Semester-IV) subjects.

*All Bridge Courses are zero (0) credit courses.

*Total and obtained marks are not added in the Grand Total.

NOIDA INSTITUTE OF ENGINEERING & TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)

Bridge Courses for Lateral Entry Students Admitted Through (B. Sc.)
B.Tech (CSBS)
EVALUATION SCHEME
SEMESTER-IV

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
WEEKS COMPULSORY INDUCTION PROGRAM													
1	ACSBS0203Z	Data Structures & Algorithms	3	1	0	30	20	50		100		150*	0
2	ACSBS0253Z	Data Structures & Algorithms Lab	0	0	4				25		25	50*	0
		GRAND TOTAL										200*	

All the students must clear the above mentioned subjects of the first year (Semester-II) Engineering Program along with the second year (Semester-IV) subjects.

*All Bridge Courses are zero (0) credit courses.

*Total and obtained marks are not added in the Grand Total.

Abbreviation Used: -

L: Lecture, T: Tutorial, P: Practical, CT: Class Test, TA: Teacher Assessment, PS: Practical Sessional, TE: Theory End Semester Exam., PE: Practical End Semester Exam.

B.TECH FIRST YEAR			
Course Code	**ACSE0101Z**	**L T P**	**Credit**
Course Title	**Problem solving using Python**	**3 0 0**	**0**

Course objective:

1	To impart knowledge of basic building blocks of Python programming
2	To provide skills to design algorithms for problem solving
3	To impart the knowledge of implementation and debugging of basic programs in Python
4	To disseminate the knowledge of basic data structures
5	To provide the knowledge of file system concepts and its application in data handling

Pre-requisites: Students are expected to be able to open command prompt window or terminal window, edit a text file, download and install software, and understand basic programming concepts.

Course Contents / Syllabus

UNIT-I	**Basics of python programming**	**8 hours**

Introduction: Introduction to computer system, algorithms, Ethics and IT policy in company, Feature of object-oriented programming, A Brief History of Python, Applications areas of python, The Programming Cycle for Python, Python IDE, Interacting with Python Programs.
Elements of Python: keywords and identifiers, variables, data types and type conversion, operators in python, expressions in python, strings.

UNIT-II	**Decision Control Statements**	**8 hours**

Conditionals: Conditional statement in Python (if-else statement, its working and execution),
Nested-if statement and elif statement in Python, Expression Evaluation & Float Representation.
Loops: Purpose and working of loops, while loop, For Loop, Nested Loops, Break and Continue, pass statement.

UNIT-III	**Function and Modules**	**8 hours**

Introduction of Function, calling a function, Function arguments, built in function, scope rules, Passing function to a function, recursion, Lambda functions
Modules and Packages: Importing Modules, writing own modules, Standard library modules, dir( ) Function, Packages in Python

UNIT-IV	**BasicData structures in Python**	**8 hours**

Strings: Basic operations, Indexing and Slicing of Strings, Comparing strings, Regular expressions.
Python Basic Data Structure: Sequence, Unpacking Sequences, Mutable Sequences, Lists, List Comprehension, Looping in lists, Tuples, Sets, Dictionaries

UNIT-V	**File and Exception handling**	**8 hours**

Files and Directories: Introduction to File Handling in Python, Reading and Writing files, Additional file methods, Working with Directories.

Exception Handling, Errors, Run Time Errors, Handling IO Exception, Try-except statement, Raise, Assert
Searching &Sorting: Simple search & Binary search, Selection Sort, Merge Sort

Course outcome: At the end of course, the student will be able to

CO 1	Write simple python programs.	K_2, K_3
CO 2	Develop python programs usingdecision control statements	K_3, K_6
CO 3	Implement user defined functions and modules in python	K_2
CO 4	Implement python data structures –lists, tuples, set, dictionaries	K_3
CO 5	Perform input/output operations with files in python and implement searching, sorting and merging algorithms	K_3, K_4

Text books

(1) Magnus Lie Hetland, "Beginning Python-From Novice to Professional"—Third Edition, Apress
(2) Python Programming using Problem solving approach by ReemaThareja OXFORD Higher education
(3) Kenneth A. Lambert, —Fundamentals of Python: First Programs, CENGAGE Learning, 2012.

Reference Books

(1) John V Guttag, —Introduction to Computation and Programming Using Python‘‘, Revised and expanded Edition, MIT Press , 2013
(2) Charles Dierbach, —Introduction to Computer Science using Python: A Computational Problem Solving Focus, Wiley India Edition, 2013.
(3) Allen B. Downey, “Think Python: How to Think Like a Computer Scientist”, 2nd edition, Updated for Python 3, Shroff/O‘Reilly Publishers, 2016
(4) Robert Sedgewick, Kevin Wayne, Robert Dondero: Introduction to Programming in Python: An Inter-disciplinary Approach, Pearson India Education Services Pvt. Ltd.,2016.
(5) Timothy A. Budd, —Exploring Python‖, Mc-Graw Hill Education (India) Private Ltd.,2015.
(6) Guido van Rossum and Fred L. Drake Jr, —An Introduction to Python – Revised and updated for Python 3.2, Network Theory Ltd., 2011.

E-book and E-Content

(1) https://www.pdfdrive.com/hacking-hacking-practical-guide-for-beginners-hacking-with-pythn-e182434771.html
(2) https://www.pdfdrive.com/python-programming-python-programming-for-beginners- python-programming-for-intermediates-e180663309.html
(3)https://www.pdfdrive.com/python-algorithms-mastering-basic-algorithms-in-the-python-language-e175246184.html
(4) https://www.pdfdrive.com/python-algorithms-mastering-basic-algorithms-in-the-python-language-e160968277.html
(5) https://docs.python.org/3/library/index.html
(6) https://www.w3schools.com/python/
(7) https://www.py4e.com/materials

Reference Links

Unit-1 https://nptel.ac.in/courses/106/106/106106182/
Unit-2 https://nptel.ac.in/courses/106/106/106106212/
Unit-3 https://nptel.ac.in/courses/106/106/106106145/
Unit-4- https://nptel.ac.in/courses/106/106/106106145/
Unit-5- https://nptel.ac.in/courses/106/106/106106145/
[Unit-2]- https://www.youtube.com/watch?v=PqFKRqpHrjw
[Unit – 3]- https://www.youtube.com/watch?v=m9n2f9lhtrw
https://www.youtube.com/watch?v=oSPMmeaiQ68
[Unit 4]- https://www.youtube.com/watch?v=ixEeeNjjOJ0&t=4s
[Unit-5]- https://www.youtube.com/watch?v=NMTEjQ8-AJM
After Completing Course Student may get certification in python using following links:
Link for Certification:
https://swayam.gov.in/nd1_noc19_cs41/preview
https://aktu.ict.iitk.ac.in/courses/python-programming-a-practical-approach/

B.TECH FIRST YEAR			
Lab Code	**ACSE0151Z**	**L T P**	**Credit**
Lab Title	Problem Solving using Python Lab	**0 0 2**	**0**

Course outcome: At the end of course, the student will be able to

CO 1	Write simple python programs.	K_2, K_3
CO 2	Implement python programs usingdecision control statements	K_3, K_6
CO 3	Writing python programs using user defined functions and modules	K_2
CO 4	Implement programs using python data structures –lists, tuples, set, dictionaries	K_3
CO 5	Write programs to perform input/output operations on files	K_3, K_4

List of Experiment:

List of Fundamental Programs

S.N.	Program Title	Category
1	Python Program to print "Hello Python"	Basic
2	Python Program to read and print values of variables of different data types.	Basic
3	Python Program to perform arithmetic operations on two integer numbers	Basic
4	Python Program to Swap two numbers	Basic
5	Python Program to convert degree Fahrenheit into degree Celsius	Operators
6	Python Program to demonstrate the use of relational operators.	Operators
7	Python Program to understand the working of bitwise and logical operators.	Operators
8	Python Program to calculate roots of a quadratic equation.	Conditional
9	Python Program to check whether a year is leap year or not.	Conditional
10	Python Program to find smallest number among three numbers.	Conditional
11	Python Program to make a simple calculator.	Conditional
12	Python Program to find the factorial of an integer number.	Loop
13	Python Program to find the reverse of an integer number.	Loop
14	Python Program to find and print all prime numbers in a list.	Loop
15	Python Program to Find the Sum of 'n' Natural Numbers	Loop
16	Python Program to print sum of series: - 1/2 + 2/3+ 3/4 + ...+n/(n+1)	Loop
17	Python Program to print pattern using nested loop	Loop
18	Python Program to Display the multiplication Table of an Integer	Loop
19	Python Program to Print the Fibonacci sequence	Loop
20	Python Program to Check Armstrong Number	Loop
21	Python Program to Find Armstrong Number in an Interval	Loop
22	Python Program to check Using function whether a passed string is palindrome or not	Function
23	Python Program using function that takes a number as a parameter, check whether the number is prime or not.	Function
24	PythonProgram using function that computes gcd of two given numbers.	Function
25	Python Program to Find LCM of two or more given numbers.	Function
26	Python Program to Convert Decimal to Binary, Octal and Hexadecimal	Function
27	Python Program To Find ASCII value of a character	Basic
28	Python Program to Display Calendar	Loop
29	Python Program to Add Two Matrices	Loop
30	Python Program to Multiply Two Matrices	Loop
31	Python Program to Transpose a Matrix	Loop
32	Python Program to Sort Words in Alphabetic Order	Sorting
33	Python Program to Display Fibonacci Sequence Using Recursion	Recursion
34	Python Program to Find Factorial of Number Using Recursion	Recursion

35	Python Program that implements different string methods.	String
36	Python Program that validates given mobile number. Number should start with 7, 8 or 9 followed by 9 digits.	String
37	Python Program to implement various methods of a list.	List
38	Python Program that has a nested list to store toppers details. Edit the details and reprint them.	List
39	Python Program to swap two values using tuple assignment.	Tuple
40	Python Program that has a set of words in English language and their corresponding Hindi words. Define dictionary that has a list of words in Hindi language and their corresponding Hindi Sanskrit. Take all words from English language and display their meaning in both languages.	Dictionary
41	Python Program that inverts a dictionary.	Dictionary
42	Python Program that reads data from a file and calculates percentage of white spaces, lines, tabs, vowels and consonants in that file.	File
43	Python Program that fetches data from a given url and write it in a file.	File
44	Python Program to understand the concept of Exception Handling	Exception Handling
45	Python Program to implement linear and binary search	Searching
46	Python Program to sort a set of given numbers using Bubble sort	Sorting

S.No.	Word Problem Experiments
1.	**String Rotation** **Problem Description** Rotate a given String in the specified direction by specified magnitude. After each rotation make a note of the first character of the rotated String, after all rotation are performed the accumulated first character as noted previously will form another string, say FIRSTCHARSTRING. Check If FIRSTCHARSTRING is an Anagram of any substring of the Original string. If yes print "YES" otherwise "NO". Input The first line contains the original string s. The second line contains a single integer q. The ith of the next q lines contains character d[i] denoting direction and integer r[i] denoting the magnitude. **Constraints** 1 <= Length of original string <= 30 1<= q <= 10 **Output** YES or NO **Explanation** **Example 1** **Input** carrace 3 L 2 R 2 L 3 **Output** NO **Explanation** After applying all the rotations, the FIRSTCHARSTRING string will be "rcr" which is not anagram of any sub string of original string "carrace".
2.	**Jurassic Park** **Problem Description** Smilodon is a ferocious animal which used to live during the Pleistocene epoch (2.5 mya–

10,000 years ago). Scientists successfully created few smilodons in an experimental DNA research. A park is established and those smilodons are kept in a cage for visitors.
This park consists of Grasslands(G), Mountains(M) and Waterbodies(W) and it has three gates (situated in grasslands only). Below is a sample layout.

W	M	G	G	G	G
M	G	W	G	M	M
G	G	G	G	G	G
W	G	G	M	W	G

Before opening the park, club authority decides to calculate Safety index of the park. The procedure of the calculation is described below. Please help them to calculate.
Safety Index calculation
Assume a person stands on grassland(x) and a Smilodon escapes from the cage situated on grassland(y). If the person can escape from any of those three gates before the Smilodon able to catch him, then the grassland(x) is called safe else it is unsafe. A person and a Smilodon both take 1 second to move from one area to another adjacent area(top, bottom, left or right) but a person can move only over grasslands though Smilodon can move over grasslands and mountains.
If any grassland is unreachable for Smilodon(maybe it is unreachable for any person also), to increase safe index value Club Authority use to mark those grasslands as safe land. Explained below

W	M	G	G	G	G
M	G	W	G(x)	M	M
G	W	G	G(y)	G	G
W	G(z)	W	M	W	G

For the above layout, there is only one gate at (4,6)
Y is the position of Smilodon's cage
X is not safe area
Z is a safe area as is it not possible for smilodon to reach z
Safety index=(total grassland areas which are safe*100)/total grassland area

Constraints

i. 3<= R,C<= 10^3
ii. Gates are situated on grasslands only and at the edge of the park
iii. The cage is also situated in grassland only
iv. The position of the cage and the position of three gates are different

Input Format

The first line of the input contains two space-separated integers R and C, denoting the size of the park (R*C)
The second line contains eight space-separated integers where
First two integers represent the position of the first gate
3rd and 4th integers represent the position of second gate
5th and 6th integers represent the position of third gate respectively
The last two integers represent the position of the cage
Next R lines, each contains space separated C number of characters. These R lines represent the park layout.

Output

Safety Index accurate up to two decimal places using Half-up Rounding method

Explanation

	Example 1 **Input** 4 4 1 1 2 1 3 1 1 3 G GGG G W W M G G W W M G M M **Output** 75.00
3.	**Bank Compare** **Problem Description** There are two banks; Bank A and Bank B. Their interest rates vary. You have received offers from both bank in terms of annual rate of interest, tenure and variations of rate of interest over the entire tenure. You have to choose the offer which costs you least interest and reject the other. Do the computation and make a wise choice. The loan repayment happens at a monthly frequency and Equated Monthly Installment (EMI) is calculated using the formula given below : EMI = loanAmount * monthlyInterestRate/(1 - 1 / (1 +monthlyInterestRate)^(numberOfYears * 12)) **Constraints** i. 1 <= P <= 1000000 ii. 1 <=T <= 50 iii. 1<= N1 <= 30 iv. 1<= N2 <= 30 **Input Format** First line : P – principal (Loan Amount) Second line : T – Total Tenure (in years). Third Line : N1 is number of slabs of interest rates for a given period by Bank A. First slab starts from first year and second slab starts from end of first slab and so on. Next N1 line will contain the interest rate and their period. After N1 lines we will receive N2 viz. the number of slabs offered by second bank. Next N2 lines are number of slabs of interest rates for a given period by Bank B. First slab starts from first year and second slab starts from end of first slab and so on. The period and rate will be delimited by single white space. **Output** Your decision – either Bank A or Bank B. **Explanation** **Example 1** **Input** 10000 20 3 5 9.5 10 9.6 5 8.5 3 10 6.9 5 8.5 5 7.9 **Output** Bank B

4.	**Cross Words** **Problem Description** A crossword puzzle is a square grid with black and blank squares, containing clue numbers (according to a set of rules) on some of the squares. The puzzle is solved by obtaining the solutions to a set of clues corresponding to the clue numbers. The solved puzzle has one letter in each of the blank square, which represent a sequence of letters (consisting of one or more words in English or occasionally other languages) running along the rows (called "Across", or "A") or along the columns (called "Down" or "D"). Each numbered square is the beginning of an Across solution or a Down solution. Some of the across and down solutions will intersect at a blank square, and if the solutions are consistent, both of them will have the same letter at the intersecting square. In this problem, you will be given the specifications of the grid, and the solutions in some random order. The problem is to number the grid appropriately, and associate the answers consistently with the clue numbers on the grid, both as Across solutions and as Down solutions, so that the intersecting blank squares have the same letter in both solutions. Rules for Clue Numbering The clue numbers are given sequentially going row wise (Row 1 first, and then row2 and so on) Only blank squares are given a clue number A blank square is given a clue number if either of the following conditions exist (only one number is given even if both the conditions are satisfied) It has a blank square to its right, and it has no blank square to its left (it has a black square to its left, or it is in the first column). This is the beginning of an Across solution with that number It has a blank square below it, and no blank square above it (it has a black square above it or it is in the first row). This is the beginning of a Down solution with that number **Constraints** i. 5<=N<=15 ii. 5<=M<=50 **Input Format** The input consists of two parts, the grid part and the solution part The first line of the grid part consists of a number, N, the size of the grid (the overall grid is N x N) squares. The next N lines correspond to the N rows of the grid. Each line is comma separated, and has number of pairs of numbers, the first giving the position (column) of the beginning of a black square block, and the next giving the length of the block. If there are no black squares in a row, the pair "0,0" will be specified. For example, if a line contains "2,3,7,1,14,2", columns 2,3,4 (a block of 3 starting with 2), 7 (a block of 1 starting with 7) and 14,15 (a block of 2 starting with 14) are black in the corresponding row. The solution part of the input appears after the grid part. The first line of the solution part contains M, the number of solutions. The M subsequent lines consist of a sequence of letters corresponding to a solution for one of the Across and Down clues. All solutions will be in upper case (Capital letters) **Output** The output is a set of M comma separated lines. Each line corresponds to a solution, and consists of three parts, the clue number, the letter A or D (corresponding to Across or Down) and the solution in to that clue (in upper case) The output must be in increasing clue number order. Ifa clue number has both an Across and a Down solution, they must come in separate lines, with the Across solution coming before the Down solution. **Explanation** **Example 1** **Input** 5

5,1
1,1,3,1,5,1
0,0
1,1,3,1,5,1
1,1
5
EVEN
ACNE
CALVE
PLEAS
EVADE

Output
1,A,ACNE
2,D,CALVE
3,D,EVADE
4,A,PLEAS
5,A,EVEN

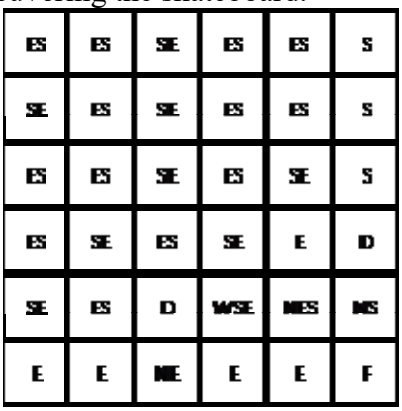
5. Skateboard

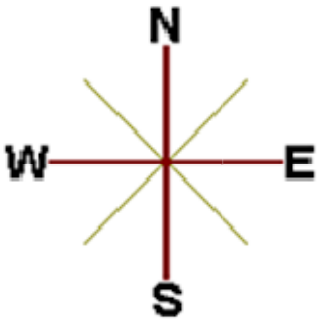
Problem Description

The amusement park at Patagonia has introduced a new skateboard competition. The skating surface is a grid of N x N squares. Most squares are so constructed with slopes that it is possible to direct the skateboard in any of up to three directions of the possible four (North ,East, South or West, represented by the letters N, E, S and W respectively). Some squares however have a deep drop from the adjacent square from which it is impossible to go to any adjacent square. These are represented by D (for Drop) in that square. The objective is to maneuver the skateboard to reach the South East corner of the grid, marked F.

Each contestant is given a map of the grid, which shows where the Drop squares are (marked D), where the Final destination is (marked F), and, for each other square, the directions it is possible to maneuver the skateboard in that square.

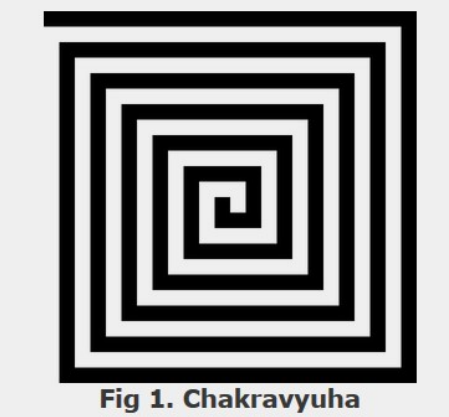
The contestant draws lots to determine which of the squares on the boundaries of the grid on the North or the West of the grid (the top or the left in the diagram) he or she should start in. Then, using a map of the grid, he or she needs to try to reach the South East corner destination by maneuvering the skateboard.





In some cases, it is impossible to reach the destination. For example, in the diagram above, if one starts at the North East corner (top right in the diagram), the only way is to go is South, until the Drop square is reached (three squares South), and the contestant is stuck there.

A contestant asks you to figure out the number of squares at the North or West boundary (top or left boundary in the map) from which it is feasible to reach the destination.

	Constraints i. 5<=N<=50 **Input Format** The first line of the input is a positive integer N, which is the number of squares in each side of the grid. The next N lines have a N strings of characters representing the contents of the map for that corresponding row. Each string may be F, representing the Final destination, D, representing a drop square, or a set of up to three of the possible four directions (N,E,S,W) in some random order. These represent the directions in which the contestant can maneuver the skateboard when in that square. **Output** The output is one line with the number of North or West border squares from which there is a safe way to maneuver the skateboard to the final destination. **Explanation** **Example 1** **Input** 6 ES,ES,SE,ES,ES,S SE,ES,SE,ES,ES,S ES,ES,SE,ES,SE,S ES,SE,ES,SE,E,D SE,ES,D,WSE,NES,NS E,E,NE,E,E,F **Output** 9
6.	**Chakravyuha** **Problem Description** During the battle of Mahabharat, when Arjuna was far away in the battlefield, Guru Drona made a Chakravyuha formation of the Kaurava army to capture YudhisthirMaharaj. Abhimanyu, young son of Arjuna was the only one amongst the remaining Pandava army who knew how to crack the Chakravyuha. He took it upon himself to take the battle to the enemies. Abhimanyu knew how to get power points when cracking the Chakravyuha. So great was his prowess that rest of the Pandava army could not keep pace with his advances. Worried at the rest of the army falling behind, YudhisthirMaharaj needs your help to track of Abhimanyu's advances. Write a program that tracks how many power points Abhimanyu has collected and also uncover his trail A Chakravyuha is a wheel-like formation. Pictorially it is depicted as below  Fig 1. Chakravyuha A Chakravyuha has a very well-defined co-ordinate system. Each point on the co-ordinate system is manned by a certain unit of the army. The Commander-In-Chief is always located at the centre of the army to better co-ordinate his forces. The only way to crack the

Chakravyuha is to defeat the units in sequential order.
A Sequential order of units differs structurally based on the radius of the Chakra. The radius can be thought of as length or breadth of the matrix depicted above. The structure i.e. placement of units in sequential order is as shown below

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

Fig 2. Army unit placements in Chakravyuha of size 5

The entry point of the Chakravyuha is always at the (0,0) co-ordinate of the matrix above. This is where the 1st army unit guards. From (0,0) i.e. 1st unit Abhimanyu has to march towards the center at (2,2) where the 25th i.e. the last of the enemy army unit guards. Remember that he has to proceed by destroying the units in sequential fashion. After destroying the first unit, Abhimanyu gets a power point. Thereafter, he gets one after destroying army units which are multiples of 11. You should also be a in a position to tell YudhisthirMaharaj the location at which Abhimanyu collected his power points.

Input Format:

First line of input will be length as well as breadth of the army units, say N

Output Format:

- Print NxN matrix depicting the placement of army units, with unit numbers delimited by (\t) Tab character
- Print Total power points collected
- Print coordinates of power points collected in sequential fashion (one per line)
- Constraints: 0 < N <=100

Sample Input and Output

S. NO.	Input	Output
1	2	1 2 4 3 Total Power points : 1 (0,0)
2	5	1 2 3 4 5 16 17 18 19 6 15 24 25 20 7 14 23 22 21 8 13 12 11 10 9 Total Power points : 3 (0,0) (4,2) (3,2)

7. Exam Efficiency

Problem Description

In an examination with multiple choice questions, the following is the exam question pattern.

- X1 number of One mark questions, having negative score of -1 for answering wrong

- X2 number of Two mark questions, having negative score of -1 and -2 for one or both options wrong
- X3 number of Three mark questions, having negative score of -1, -2 and -3 for one, two or all three options wrong
- Score Required to Pass the exam : Y
- For 1,2 and 3 mark questions, 1,2 and 3 options must be selected. Simply put, once has to attempt to answer all questions against all options.

Identify the minimum accuracy rate required for each type of question to crack the exam. Calculations must be done up to 11 precision and printing up to 2 digit precision with ceil value

Input Format:

First line contains number of one mark questions denoted by X1,
Second line contains number of two mark questions denoted by X2
Third line contains number of three mark questions denoted by X3
Fourth line contains number of marks required to pass the exam denoted by Y.

Output Format:

Minimum Accuracy rate required for one mark question is 80%
Minimum Accuracy rate required for Two mark question is 83.33%
Minimum Accuracy rate required for Three mark question is 90%

Note: - If the mark required to pass the exam can be achieved by attempting without attempting any particular type of question then show message similar to, One mark question need not be attempted, so no minimum accuracy rate applicable

Sample Input and Output

S.No.	Input	Output	Explanation
1	20 30 30 120	One mark questions need not be attempted, so no minimum accuracy rate applicable. Minimum Accuracy rate required for Two mark question is 58.33% Minimum Accuracy rate required for Three mark question is 72.23%	If one got full marks in two marks question and three marks question then total accuracy can be 0 in one mark question In same way it will be done for two marks and three marks question
2	20 30 30 170	Minimum Accuracy rate required for one mark question is 100% Minimum Accuracy rate required for Two mark question is 100% Minimum Accuracy rate required for Three mark question is 100%	If one got full marks in two marks question and three marks question then total accuracy should be 100% in one mark question to pass the exam. In same way it will be done for two marks and three marks question

8. Calculate Salary and PF

Problem Description

Calculate the Final Salary & Final Accumulated PF of an Employee working in ABC Company Pvt. Ltd. The Company gives two Increments (i.e. Financial Year Increment & Anniversary Increment) to an Employee in a Particular Year.

The Employee must have Completed 1 Year to be Eligible for the Financial Year Increment. The Employee who are joining in the month of Financial Year Change (i.e. April) are

considered as the Luckiest Employee's, because after completion of 1 Year, they get Two Increments
(Financial Year Increment & Anniversary Increment).
Rate of Interest for the Financial Year Increment = 11%.
Rate of Interest for the Anniversary Increment = 12%.
From 4th Year, the Financial Year Increment will be revised to 9%.
From 8th Year, the Financial Year Increment will be revised to 6%.
The Company is giving special Increment for the Employee who have completed 4 years & 8 years respectively.
So, the Anniversary Increment of the Employee for the 4th Year will be 20% and the Anniversary Increment of the Employee for the 8th year will be 15%.
Calculate the Final Salary after N number of Years as well as Calculate the Accumulated PF of the Employee after N number of Years.
Please Note that, the Rate of Interest for calculating PF for a Particular Month is 12%. Moreover, take the upper Limit of the amount if it is in decimal (For e.g. - If any Amount turns out to be 1250.02, take 1251 for the Calculation.)

Input Format:

i. Joining Date in dd/mm/yy format
ii. Current CTC.
iii. Number of Years for PF & Salary Calculation.

Output Format:

i. Salary after the Specified Number of Years (i.e. CTC after N number of Years) in the following format
Final Salary =
ii. Accumulated PF of the Employee after N number of Years in the following format
Final Accumulated PF =

Constraints:

Calculation should be done upto 11-digit precision and output should be printed with ceil value

Sample Input and Output

S.No.	Input	Output
1	5 01/01/2016 10000 2	Final Salary = 13924 Final Accumulated PF = 2665
2	19/01/2016 6500 4	Final Salary = 14718 Final Accumulated PF = 4343

9. ISL Schedule

Problem Description

The Indian Soccer League (ISL) is an annual football tournament.
The group stage of ISL features N teams playing against each other with following set of rules:

i. N teams play against each other twice - once at Home and once Away
ii. A team can play only one match per day
iii. A team cannot play matches on consecutive days
iv. A team cannot play more than two back to back Home or Away matches
v. Number of matches in a day has following constraints
a. The match pattern that needs to be followed is -

- Day 1 has two matches and Day 2 has one match,
- Day 3 has two matches and Day 4 has one match and so on

b. There can never be 3 or more matches in a day

vi. Gap between two successive matches of a team cannot exceed floor(N/2) days where floor is the mathematical function floor()

vii. Derby Matches (any one)

a. At least half of the derby matches should be on weekend

b. At least half of the weekend matches should be derby matches

Your task is to generate a schedule abiding to above rules.

Input Format:

First line contains number of teams (N).
Next line contains state ID of teams, delimited by space

Output Format:

Match format: Ta-vs-Tb
where Ta is the home team with id a and Tb is the away team with id b.
For each day print the match(es) in following format:-
Two matches:- "#D Ta-vs-Tb Tm-vs-Tn"
One match:- "#D Tx-vs-Ty"
where D is the day id and [a, b, m, n, x, y] are team ids.

Constraints:

i. 8 <= N <= 100

Note :

- Team ids are unique and have value between 1 to N
- Day id starts with 1
- Every 6th and 7th day are weekends
- Derby is a football match between two teams from the same state

Sample Input and Output

S.No.	Input	Output
1	8 1 2 5 4 3 1 6 6	#1 T1-vs-T6 T3-vs-T5 #2 T7-vs-T4 #3….and so on

Note: - There can be multiple correct answers for the same test cases. For better understanding of test case refer this PDF. This PDF contains one of the correct answer for a test case.

Explanation:

There are 8 teams with following information: -

Team ID	1	2	3	4	5	6	7	8
State ID	1	2	5	4	3	1	6	6

10. Longest Possible Route

Problem Description

Given an MxN matrix, with a few hurdles arbitrarily placed, calculate the cost of longest possible route from point A to point B within the matrix.

Input Format:

i. First line contains 2 numbers delimited by whitespace where, first number M is number of rows and second number N is number of columns

ii. Second line contains number of hurdles H followed by H lines, each line will

contain one hurdle point in the matrix.

iii. Next line will contain point A, starting point in the matrix.

iv. Next line will contain point B, stop point in the matrix.

Output Format:

Output should display the length of the longest route from point A to point B in the matrix.

Constraints:

i. The cost from one position to another will be 1 unit.

ii. A location once visited in a particular path cannot be visited again.

iii. A route will only consider adjacent hops. The route cannot consist of diagonal hops.

iv. The position with a hurdle cannot be visited.

v. The values MxN signifies that the matrix consists of rows ranging from 0 to M-1 and columns ranging from 0 to N-1.

vi. If the destination is not reachable or source/ destination overlap with hurdles, print cost as -1.

Sample Input and Output

S. No.	Input	Output	Explanation
1	3 10 3 1 2 1 5 1 8 0 0 1 7	24	Here matrix will be of size 3x10 matrix with a hurdle at (1,2),(1,5) and (1,8) with starting point A(0,0) and stop point B(1,7) 3 10 3 -- (no. of hurdles) 1 2 1 5 1 8 0 0 -- (position of A) 1 7 -- (position of B) (->) count is 24. So final answer will be 24. No other route longer than this one is possible in this matrix.
2	2 2 1 0 0 1 1 0 0	-1	No path is possible in this 2*2 matrix so answer is -1

11. Min Product array

Problem Description

The task is to find the minimum sum of Products of two arrays of the same size, given that k modifications are allowed on the first array. In each modification, one array element of the first array can either be increased or decreased by 2.

Note- the product sum is Summation (A[i]*B[i]) for all i from 1 to n where n is the size of both arrays

Input Format:

i. First line of the input contains n and k delimited by whitespace

ii. Second line contains the Array A (modifiable array) with its values delimited by spaces

iii. Third line contains the Array B (non-modifiable array) with its values

delimited by spaces

Output Format:
Output the minimum sum of products of the two arrays
Constraints:

i. $1 \le N \le 10^5$
ii. $0 \le |A[i]|, |B[i]| \le 10^5$
iii. $0 \le K \le 10^9$

Sample Input and Output

S.No.	Input	Output
1	3 5 1 2 -3 -2 3 -5	-31
2	5 3 2 3 4 5 4 3 4 2 3 2	25

Explanation for sample 1:
Here total numbers are 3 and total modifications allowed are 5. So we modified A[2], which is -3 and increased it by 10 (as 5 modifications are allowed). Now final sum will be
(1 * -2) + (2 * 3) + (7 * -5)
-2 + 6 - 35
-31
-31 is final answer.
Explanation for sample 2:
Here total numbers are 5 and total modifications allowed are 3. So we modified A[1], which is 3 and decreased it by 6 (as 3 modifications are allowed).
Now final sum will be
(2 * 3) + (-3 * 4) + (4 * 2) + (5 * 3) + (4 * 2)
6 - 12 + 8 + 15 + 8
25
25 is final answer.

12. **Consecutive Prime Sum**

Problem Description

Some prime numbers can be expressed as a sum of other consecutive prime numbers. For example, 5 = 2 + 3, 17 = 2 + 3 + 5 + 7, 41 = 2 + 3 + 5 + 7 + 11 + 13. Your task is to find out how many prime numbers which satisfy this property are present in the range 3 to N subject to a constraint that summation should always start with number 2.
Write code to find out the number of prime numbers that satisfy the above-mentioned property in a given range.

S. No.	Input	Output	Comment
1	20	2	(Below 20, there are 2 such members: 5 and 17) 5 = 2+3 17 = 2+3+5+7
2	15	1	

Input Format:
First line contains a number N
Output Format:
Print the total number of all such prime numbers which are less than or equal to N.

	Constraints: 2<N<=12,000,000,000
13.	**kth largest factor of N** **Problem Description** A positive integer d is said to be a factor of another positive integer N if when N is divided by d, the remainder obtained is zero. For example, for number 12, there are 6 factors 1, 2, 3, 4, 6, 12. Every positive integer k has at least two factors, 1 and the number k itself.Given two positive integers N and k, write a program to print the kth largest factor of N. **Input Format:** The input is a comma-separated list of positive integer pairs (N, k) **Output Format:** The kth highest factor of N. If N does not have k factors, the output should be 1. **Constraints:** 1<N<10000000000. 1<k<600.You can assume that N will have no prime factors which are larger than 13. **Example 1** **Input:** 12,3 **Output:** 4 **Explanation:** N is 12, k is 3. The factors of 12 are (1,2,3,4,6,12). The highest factor is 12 and the third largest factor is 4. The output must be 4
14.	**Coins Distribution Question (or Coins Required Question)** **Problem Description** Find the minimum number of coins required to form any value between 1 to N, both inclusive. Cumulative value of coins should not exceed N. Coin denominations are 1 Rupee, 2 Rupee and 5 Rupee. Let's understand the problem using the following example. Consider the value of N is 13, then the minimum number of coins required to formulate any value between 1 and 13, is 6. One 5 Rupee, three 2 Rupee and two 1 Rupee coins are required to realize any value between 1 and 13. Hence this is the answer. However, if one takes two 5 Rupee coins, one 2 rupee coins and two 1 rupee coins, then to all values between 1 and 13 are achieved. But since the cumulative value of all coins equals 14, i.e., exceeds 13, this is not the answer. **Input Format** A single integer value **Output Format** Four Space separated Integer Values 1st – Total Number of coins 2nd – number of 5 Rupee coins. 3rd – number of 2 Rupee coins. 4th – number of 1 Rupee coins. **Constraints** 0<n<1000 **Sample Input:** 13 **Sample Output:** 6 1 3 2
S. NO.	**Debugging Experiments**
1.	Write error/output in the following code.

	# abc.py deffunc(n): return n + 10 func('Hello')

2.	Write the output of the following code.
	
	if not a or b:
	  print 1
	elif not a or not b and c:
	  print 2
	elif not a or b or not b and a:
	  print 3
	else:
	  print 4

3.	Write error/output in the following code.
	
	count = 1
	
	defdoThis():
	
	  global count
	
	  for i in (1, 2, 3):
	  count += 1
	
	doThis()
	
	print count

4.	Write the output of the following code.
	
	check1 = ['Learn', 'Quiz', 'Practice', 'Contribute']
	check2 = check1
	check3 = check1[:]
	
	check2[0] = 'Code'
	check3[1] = 'Mcq'
	
	count = 0
	for c in (check1, check2, check3):
	  if c[0] == 'Code':
	  count += 1
	  if c[1] == 'Mcq':
	  count += 10
	
	print count

5.	What is the output of the following program?
	
	D = dict()
	for x in enumerate(range(2)):
	  D[x[0]] = x[1]
	  D[x[1]+7] = x[0]

	print(D)
6.	What is the output/error in the following program? D = {1 : 1, 2 : '2', '1' : 1, '2' : 3} D['1'] = 2 print(D[D[D[str(D[1])]]])
7.	What is the output/error in the following program? D = {1 : {'A' : {1 : "A"}, 2 : "B"}, 3 :"C", 'B' : "D", "D": 'E'} print(D[D[D[1][2]]], end = " ") print(D[D[1]["A"][2]])

8.	What is the output/error in the following program?
	
	D = dict()
	for i in range (3):
	  for j in range(2):
	  D[i] = j
	print(D)

9.	What is the output/error in the following program? x = ['ab', 'cd'] for i in x: x.append(i.upper()) print(x)

10.	What is the output/error in the following program?
	
	i = 1
	while True:
	  if i%3 == 0:
	  break
	  print(i)
	i + = 1

B.TECH FIRST YEAR					
Course Code	**ACSBS0103Z**	**L**	**T**	**P**	**Credit**
Course Title	**Fundamentals of Computer Science**	**3**	**0**	**0**	**0**

Course objective:

The course covers various operations, conditional statements and looping constructs in C. The course aims to solve complex problems using functions and arrays in C.

Pre-requisites: **Basic Knowledge of Computer**

Course Contents / Syllabus

UNIT-I	**General problem Solving concepts**	**5 hours**

Algorithm, and Flowchart for problem solving with Sequential Logic Structure, Decisions and Loops.
Programming using C: applications of C programming, Structure of C program, Overview of compilation and execution process in an IDE, transition from algorithm to program, Syntax, logical errors and Run time errors, object and executable code

UNIT-II	**Imperative languages&Operators**	**7 hours**

Introduction to imperative language; syntax and constructs of a specific language (ANSI C)
Types Operator and Expressions with discussion of variable naming and Hungarian Notation: Variable Names, Data Type and Sizes (Little Endian Big Endian), Constants, Declarations, Arithmetic Operators, Relational Operators, Logical Operators, Type Conversion, Increment Decrement Operators, Bitwise Operators, Assignment Operators and Expressions, Precedence and Order of Evaluation, proper variable naming and Hungarian Notation.

UNIT-III	**Control Flow**	**6 hours**

Control Flow with discussion on structured and unstructured programming: Statements and Blocks, If-Else-If, Switch, Loops – while, do, for, break and continue, goto labels, structured and un- structured programming.

UNIT-IV	**Functions and Program Structure**	**8 hours**

Functions and Program Structure with discussion on standard library: Basics of functions, parameter passing and returning type, C main return as integer, External, Auto, Local, Static, Register Variables, Scope Rules, Block structure, Initialization, Recursion, Pre-processor, Standard Library Functions and return types.

UNIT-V	**Pointers and Arrays**	**8 hours**

Pointers and address, Pointers and Function Arguments, Pointers and Arrays, Address Arithmetic, character Pointers and Functions, Pointer Arrays, Pointer to Pointer, Multi-dimensional array and Row/column major formats, Initialization of Pointer Arrays, Command line arguments, Pointer to functions, complicated declarations and how they are evaluated.
Structures: Basic Structures, Structures and Functions, Array of structures, Pointer of structures, Self-referral structures, Table look up, typedef, unions, Bit-fields

UNIT-VI	**Input and Output:**	**6 Hours**

Standard I/O, Formatted Output – printf, Formated Input – scanf, Variable length argument list, file access including FILE structure, fopen, stdin, stdout and stderr, Error Handling including exit, perror and error.h, Line I/O, related miscellaneous functions.
Unix system Interface: File Descriptor, Low level I/O – read and write, open, create, close and unlink, Random access – seek, Discussions on Listing Directory, Storage allocator.
Programming Method: Debugging, Macro, User Defined Header, User Defined Library Function, makefile utility

Course outcome: At the end of course, the student will be able to

CO 1	Acquire a broad perspective about the uses of computers in engineering industry.	K2

CO 2	Understand the concept of computers, algorithm and algorithmic thinking.	K2
CO 3	Apply conditional statements and looping constructs.	K3
CO 4	Implement array and perform operations on it.	K3
CO 5	Understand the more advanced features of the C language	K2

Text Books

1. B. W. Kernighan and D. M. Ritchi, The C Programming Language, 1988, 2nd Edition, PHI.
2. B. Gottfried, Programming in C, Schaum Outline Series, 1996, 2nd Edition, McGraw Hill Companies Inc.

Reference Books

1. Herbert Schildt, C: The Complete Reference, 2000, 4th edition, McGraw Hill.
2. YashavantKanetkar, Let Us C, 2017, 15th edition, BPB Publications.

B.TECH FIRST YEAR			
Course Code	**ACSBS0153Z**	**L T P**	**Credit**
Course Title	**Fundamentals of Computer Science Lab**	**0 0 4**	**0**

Suggested List of Experiments	**CO**
1. Algorithm and flowcharts of small problems like GCD	1
2. Structured code writing with:	1
i. Small but tricky codes	1
ii. Proper parameter passing	1
iii. Command line Arguments	1
iv. Variable parameter	2
v. Pointer to functions	2
vi. User defined header	3
vii. Make file utility	3
viii. Multi file program and user defined libraries	4
ix. Interesting substring matching / searching programs	4
x. Parsing related assignments	4

Lab Course Outcome:

CO 1	Read, understand and trace the execution of programs written in C language.	K2
CO 2	Write the C code for a given algorithm.	K2
CO 3	Implement Programs with pointers and arrays, perform pointer arithmetic, and use the pre-processor.	K3
CO 4	Write programs that perform operations using derived data types.	K2
CO5	Implement String Handling	K3

B.TECH FIRST YEAR			
Course Code	**ACSE0202Z**	**L T P**	**Credit**
Course Title	**Problem solving using Advanced Python**	**3 1 0**	**0**

Course objective: The objective of the course is to make its students able

1	To learn the Object Oriented Concepts in Python
2	To learn the concept of reusability through inheritance and polymorphism
3	To impart the knowledge of functional programming
4	To learn the concepts of designing graphical user interfaces
5	To explore the knowledge of standard Python libraries

Pre-requisites: Students are expected to have basic knowledge of programming concepts of python programming.

Course Contents / Syllabus

UNIT-I	**Classes and Objects**	**8 hours**

Introduction: Python Classes and objects, User-Defined Classes, Encapsulation, Data hiding , Class Variables and Instance Variables, Instance methods, Class method, static methods, constructor in python, parametrized constructor, Magic Methods in python, Object as an argument, Instances as Return Values, namespaces

UNIT-II	**Object Oriented Concepts**	**8 hours**

Introduction to the Specialization, Inheritance, Types of inheritance, Invoking the Parent Class's Method, Method overriding, abstract class, MRO and super ( ), Polymorphism
Introspection: Introspecting types, Introspecting objects, Introspecting scopes, inspect modules, introspect tools

UNIT-III	**Functional Programming**	**8 hours**

Map, filter, Reduce, Comprehensions, Immutability, Closures and Decorators, generators, Co-routines, iterators, Declarative programming

UNIT-IV	**GUI Programming**	**8 hours**

Ipywidgets Package, Numeric Widgets, Boolean Widgets, Selection Widgets, String Widgets, Date Picker, Color Picker, Container Widgets, Creating a GUI Application, Tkinter, button, canvas.

UNIT-V	**Libraries in Python**	**8 hours**

NumPy: Basic Operation, Indexing, slicing and Iterating, multidimensional arrays, NumPy Data types, Reading and writing data on Files,**Pandas :** Series and Data Frames, Grouping, aggregation, Merge Data Frames, Generate summary tables, Group data into logical pieces, Manipulation of data. **SciPy:** Introduction to SciPy, Create function, modules of SciPy. **Matplotlib:** Scatter plot, Bar charts, histogram, Stack charts, Legend title Style, Figures and subplots, Plotting function in pandas, Labelling and arranging figures, Save plots. **Seaborn:** style function, color palettes, distribution plots, category plot, regression plot.

Course outcome: At the end of course, the student will be able to

CO 1	Define classes and create instances in python	K_1, K_2
CO 2	Implement concept of inheritance and polymorphism using python	K_3
CO 3	Implement functional programming in python	K_2

CO 4	Create GUI based Python application	K_3
CO 5	Applythe concept of Python libraries to solve real world problems	K_3, K_6

Text books

(1) Magnus Lie Hetland, "Beginning Python-From Novice to Professional"—Third Edition, Apress

(2) Peter Morgan, Data Analysis from Scratch with Python, AI Sciences

(3) Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", 2nd edition, Updated for Python 3, Shroff/O'Reilly Publishers, 2016

(4) Miguel Grinberg, Developing Web applications with python, OREILLY

Reference Books

(1) Dusty Phillips, Python 3 Object-oriented Programming - Second Edition, O'Reilly

(2) Burkhard Meier, Python GUI Programming Cookbook - Third ,Packt

(3) DOUG HELLMANN, THE PYTHON 3 STANDARD LIBRARY BY EXAMPLE, :Pyth 3 Stan Libr Exam _2 (Developer's Library) 1st Edition, Kindle Edition.

(4) Kenneth A. Lambert, ―Fundamentals of Python: First Programs‖, CENGAGE Learning, 2012.

E-books& E-Contents:

(1)https://www.pdfdrive.com/a-python-book-beginning-python-advanced-python-and-python- exercises-e125280.html

(2)https://www.pdfdrive.com/a-python-book-beginning-python-advanced-python-and-python-e9236005.html

(3)https://www.pdfdrive.com/learn-python-in-one-day-and-learn-it-well-python-for-beginners-with-hands-on-project-the-only-book-you-need-to-start-coding-in-python-immediately-e183833259.html

(4)https://www.pdfdrive.com/python-programming-python-programming-for-beginners-python-programming-for-intermediates-d180663309.html

(5)https://www.pdfdrive.com/python-programming-python-programming-for-beginners-python-programming-for-intermediates-d180663309.html

(6) https://realpython.com/tutorials/advanced/

Reference Links

Unit 1-https://nptel.ac.in/courses/106/106/106106145/

Unit-2-https://www.python-course.eu/python3_inheritance.php

Unit -3 https://realpython.com/courses/functional-programming-python/

Unit-4: https://realpython.com/python-gui-tkinter/

Unit-5: https://nptel.ac.in/courses/106/107/106107220/
https://nptel.ac.in/courses/106/106/106106212/
https://nptel.ac.in/courses/106/105/106105152/

https://www.youtube.com/watch?v=98YeQpmQeH8

https://www.youtube.com/watch?v=u9x475OGj_U

https://www.youtube.com/watch?v=HFW7eA9wUxY

https://www.youtube.com/watch?v=byHcYRpMgI4

https://www.youtube.com/watch?v=9N6a-VLBa2I

https://www.youtube.com/watch?v=Ta1bAMOMFOI

https://www.youtube.com/watch?v=FsAPt_9Bf3U

https://www.youtube.com/watch?v=LwPTfwlry1s

https://www.youtube.com/watch?v=YXPyB4XeYLA

https://www.youtube.com/watch?v=dVr7r7QgLrk&t=21s

Students may follow Links given below to get certification in course of Advanced python

Link for Certification in Python
https://swayam.gov.in/nd1_noc20_cs36/preview
https://swayam.gov.in/nd1_noc20_cs46/preview

B.TECH FIRST YEAR			
Lab Code	**ACSE0252Z**	**L T P**	**Credit**
Lab Title	**Problem Solving using Advanced Python Lab**	**0 0 2**	**0**

Course outcome:At the end of course, the student will be able to

CO 1	Write programs to create classes and instances in python	K_1, K_3
CO 2	write programs to Implement concept of inheritance and polymorphism using python	K_2, K_3
CO 3	Write programs using functional programming in python	K_4
CO 4	write programs to create GUI based Python application	K_3, K_4
CO 5	Developing real life applications using python libraries to solve real world problems	K_4, K_6

List of Experiment :

S.No.	Name of Experiment	
	Class and Methods	
1	Python program to demonstrate instantiating a class.	
2	Python program to demonstrate use of class method and static method	
3	Python program to implement constructors.	
4	Python program to show that the variables with a value assigned in the class declaration, are class variables and variables inside methods and constructors are instance variables.	
5	Python program to create Bank-account class with deposit, withdraw function	
	Inheritance	
6	Python program to demonstrate single inheritance	
7	Python program to demonstrate multilevel inheritance	
8	Python program to demonstrate multiple inheritance	
9	Python program to demonstrate hierarchical inheritance	
10	Python program to demonstrate hybrid inheritance	
	Polymorphism	
11	Python program to demonstrate in-built polymorphic function	
12	Python program to demonstrate user defined polymorphic functions	
13	Python program to demonstrate method overriding	
	Functional Programming	
14	Python program to demonstrate working of map	
15	Python program to demonstrate working of filter	
16	Python program to demonstrate working of reduce	
17	Python program to demonstrate immutable data types	
18	Python program to demonstrate Monkey Patching in Python	
19	Python program to demonstrate decorators with parameters in python	
20	Python program to demonstrate conditional decorators	
21	Python program to demonstrate nested decorators	
22	Python program to demonstrate chain multiple decorators	

23	Python program to demonstrate use of generators	
24	Python program to demonstrate working of iterators	
25	Write a Python program to create a table and insert some records in that table. Finally selects all rows from the table and display the records.	
	GUI Programming	
26	Python Program to understand working of various Tkinter widgets	
27	Create a Distance-time GUI calculator using Tkinter	
28	Write a NumPy program to calculate the difference between the maximum and the minimum values of a given array along the second axis.	
29	Write a Python program to create a 2-D array with ones on the diagonal and zeros elsewhere. Now convert the NumPy array to a SciPy sparse matrix in CSR format.	
30	Write a Python program to add, subtract, multiple and divide two Pandas Series.	
31	Write a program to Create Your Plot using python. Also add and delete axes.	
32	Write a program to plot data using seaborn and show the plot.	

B.TECH FIRST YEAR			
Course Code	**ABT0201Z**	**L T P**	**Credit**
Course Title	**Introduction to Biotechnology**	**3 0 0**	**0**

Course objective: 1. To develop a basic understanding of biotechnology.
2. To provide an overview of cell biology, microbiology and biotechnological advancements

Pre-requisites: Students should know about basic concept of biology

Course Contents / Syllabus

UNIT-I	**Biochemistry**	
Component of the cell, structure and biochemical functions, Biomolecules-Carbohydrates, lipids, proteins, Nucleic acids, Structure and classification of enzymes		
UNIT-II	**Cell Biology and Microbiology**	
Eukaryotic, Prokaryotic cells, Cell cycle – Mitosis and Meiosis, History and development of Microbiology, Classification and Nomenclature of Microorganisms - concept of kingdom-protista, prokaryote and eukaryotes		
UNIT-III	**Molecular Biology**	
Introduction to nucleic acids: Nucleic acids as genetic material, Structure and physicochemical properties of elements in DNA and RNA, Biological significance of differences in DNA and RNA.		
UNIT-IV	**Immunology**	
Cells of immune system, Development, maturation, activation and differentiation of T-cells and B-cells, Phagocytosis process		
UNIT-V	**Biotechnology Applications**	
Industrial production, Drug discovery and development, applications of biotechnology include GMO (genetically modified organism), biopesticides, insulin, gene therapy, transgenic animals, bioremediation, biotechnology sector in India		

Course outcome: **After completion of this course students will be able to**

CO 1	Acquire the basic knowledge of biomolecules and their functions.	
CO 2	Understand the concept of cell structure and microbiology.	
CO 3	Understand the concept of nucleic acids and their key functions	
CO 4	Understand the concept of immune system and various components involved in it.	
CO 5	Describe the wide applications of biotechnology and concept of bioinformatics.	

Text books (Atleast3)

1. Introduction To Biotechnology 3rd Edition by Thieman and William, Pearson
2. Biotechnology by BD Singh. Kalyani Publishers.

Reference Books (Atleast 3)

1. Biology 12th Edition by Raven and George Johnson and Kenneth Mason and Jonathan Losos and Tod Duncan. McGrawHill Publications
2. TEXTBOOK OF BIOTECHNOLOGY by PATNAIK, McGraw Hill
3. Basic Biotechnology3rd Edition by Colin Ratledge&Bjorn Kristiansen, Cambridge University Press

NPTEL/ Youtube/ Faculty Video Link:

Unit 1	https://www.youtube.com/watch?v=DhwAp6yQHQI https://www.youtube.com/watch?v=f7jRpniCsaw
Unit 2	https://www.youtube.com/watch?v=Bhe6Tj2Ebys
Unit 3	https://www.youtube.com/watch?v=jLyi2K-29xU
Unit 4	https://www.youtube.com/watch?v=Dyv6YiH5rME
Unit 5	https://www.youtube.com/watch?v=2zLn-RngMU4

B.TECH FIRST YEAR			
Course Code	**ABT0251Z**	**L T P**	**Credit**
Course Title	**Introduction to Biotechnology Lab**	**0 0 2**	**0**

Suggested list of Experiment

Sr. No.	Name of Experiment	CO
1	Estimation of carbohydrates	1
2	Preparation and study of mitosis in onion root tips.	1
3	Mitotic and meiotic studies in grasshopper testes	1
4	Preparation and sterilization of equipment and culture media.	1
5	Enumeration of bacteria from soil samples.	1
6	Demonstration of agarose gel electrophoresis for DNA visualization.	1
7	Introduction to types of sequence databases (Nucleotide & Protein)	2
8	Retrieving sequences from the databases	2

Lab Course Outcome: After completion of this course students will be able to:

CO 1	Understand the basic techniques of biochemistry, microbiology and cell biology
CO 2	Understand the applications of biotechnology and bioinformatics.

B.TECH FIRST YEAR			
Course Code	**ACSE0201Z**	**L T P**	**Credit**
Course Title	**Programming for Problem Solving using C**	**3 1 0**	**0**

Course objective: The objective of the course is to make its students able

1	To understand basic concepts of C-programming language
2	To implement C programs to solve complex problems
3	To enhance debugging, analysing and problem-solving skills
4	To create diversified solutions for real world applications using C language
5	To acquire the knowledge of variable allocation andbinding, conditional statement, control flow, types, function, pointer, parameter passing, array, structure and file handling to solve real world problems

Pre-requisites: Students are expected to be able to open command prompt window or terminal window, edit a text file, download and install software, and understand basic programming concepts.

Course Contents / Syllabus

UNIT-I	Basic concepts	8hours

Introduction to components of a computer system: Memory, processor, I/O Devices, operating system, Concept of Assembler, compiler, interpreter, linker and loader.
Number System: introduction to number system, binary arithmetic.
Concept of algorithms, Flow Charts.

UNIT-II	Introduction to Programming	8 hours

Programming using C:applications of C programming, Structure of C program, Overview of compilation and execution process in an IDE, transition from algorithm to program, Syntax, logical errors and Run time errors, object and executable code,Tokens of C language: Keywords, identifiers, constant, data types.

Arithmetic expressions and precedence: Operators,operator precedence and associativity, type conversion, mixed operands, Pitfalls/Issues with sizeof () usage.

UNIT-III	Decision Control Statements, pre-processor directives	8 hours

Conditional Branching: if, else-if, nested if - else, switch statements, use of break and default with switch.
Iteration and loops:Concept of loops, for, while and do-while, multiple loop variables, use of break and continue statements, nested loop.
Pre-processor directives: defining and calling macros, file inclusion, conditional compilation.
Pointers: defining and declaring pointer, pointer arithmetic and scaling, Pointer Aliasing.

UNIT-IV	Functions and Arrays	8 hours

Functions: Concept of Sub-programming, function, types of functions, passing parameters to functions: call by value, call by reference, recursive functions, scope of variable, local and global variables, Nesting of Scope, Storage classes: Auto, Register, Static and Extern

Arrays: Array notation and representation (one and two dimensional), array using pointers, manipulating array elements, 2-d arrays used in matrix computation. Strings and C string library, Structure, union, Array of structures, Self-referential structures, passing arrays and structure as arguments
Searching techniques (Linear, Binary Search), Sorting Algorithms (Bubble, Insertion and Selection)
Introduction to dynamic memory allocation (malloc(), calloc(), realloc(), free())

UNIT-V	File handling and Introduction to Embedded Programming	8 hours

File handling: File Pointer, File I/O functions and modes, Input and Output using file pointers, Character

Input and Output with Files.
Introduction to Embedded Programming: Embedded systems, Introduction to 8051microcontrolller, Installing the Keil software and loading the project, Configuring the simulator, Building the target, Running the simulation, Dissecting the program.
Case Study: Intruder Alarm System.

Course outcome: At the end of course, the student will be able to

CO 1	Develop simple algorithms for arithmetic and logical problems.	K_2
CO 2	Implement and trace the execution of programs written in C language.	K_1, K_2, K_4
CO 3	Implement conditional branching and iteration	K_3
CO 4	Use function, arrays and structures to develop algorithms and programs.	K_2, K_6
CO 5	Use searching and sorting algorithm to arrange data and use file handling for developing real life projects	K_2, K_4

Textbooks:

(1) Herbert Schildt, "C: The Complete Reference", OsbourneMcGraw Hill, 4th Edition, 2002.

(2) E Balaguruswami, "Computer Concepts and Programming in C", McGraw Hill, 2010.

(3) Michael J. Pont, "Embedded C", Addison-wesley Pearson Education, 2002.

Reference Books:

(1) The C programming by Kernighan Brain W. and Ritchie Dennis M., Pearson Education.

(2) Yashwant P. Kanetkar"Let Us C", BPB publication, 2017.

(3) Computer Basics and C Programming by V. Rajaraman, PHI Learning pvt. Limited, 2015.

(4) Yashwant P. Kanetkar, "Working with C", BPB publication, 2003.

E-Book Links:

(1) https://en.wikibooks.org/wiki/C_Programming

(2) https://en.wikibooks.org/wiki/A_Little_C_Primer

(3) https://www.goodreads.com/book/show/6968572-ansi-c-programming

(4)https://www.pdffiller.com/347652461-projects-in-c-by-yashwant-kanetkar-pdfpdf-c-projects-yashwant-kanetkar-pdf-form-

(5)http://www.freebookcentre.net/programming-books-download/Lecture-Notes-On-C-Programming-by-L.-V.-Narasimha-Prasad-and-E.-Krishnarao-Patro.html

Reference Links:

(1) https://nptel.ac.in/courses/106/104/106104128/

(2)https://nptel.ac.in/courses/106/104/106104074/

(3)https://nptel.ac.in/courses/106/102/106102066/

(4)https://nptel.ac.in/courses/106/105/106105171/

(5)https://www.youtube.com/watch?v=IdXrCPzNnkU&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=4

(6)https://www.youtube.com/watch?v=L2oataK7F10&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=11

(7)https://www.youtube.com/watch?v=K538VFFmFGc&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=14
(8)https://www.youtube.com/watch?v=HyDpW7Al6_E&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=15
(9)https://www.youtube.com/watch?v=0g82dDC-mtc&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=17
(10)https://www.youtube.com/watch?v=d1EHD8RoLDQ&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=19
(11)https://www.youtube.com/watch?v=5xJ1GXTa7IU&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=21
(12)https://www.youtube.com/watch?v=I9828WOCEMg&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=26
(13)https://www.youtube.com/watch?v=V7AZuMuJmXY&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=32
(14)https://www.youtube.com/watch?v=AJvCmpt1UU8&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=37
(15)https://www.youtube.com/watch?v=1iwmwEJhcMw&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=39
(16)https://www.youtube.com/watch?v=K4qXMLItABI&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=45
(17)https://www.youtube.com/watch?v=LoIe_9cTtPE&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=53
(18)https://www.youtube.com/watch?v=kDDd7AmXq1w&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=55
(19)https://www.youtube.com/watch?v=Z_0xXmOgYtY&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=58
(20)https://www.youtube.com/watch?v=u60YRSB2isQ&list=PLJ5C_6qdAvBFzL9su5J-FX8x80BMhkPy1&index=61

B.TECH FIRST YEAR			
Lab Code	**ACSE0251Z**	**L T P**	**Credit**
Lab Title	Programming for Problem Solving Using C Lab	**0 0 2**	**0**

Course outcome: At the end of course, the student will be able to

CO 1	Write programs for arithmetic and logical problems.	K_1, K_3
CO 2	write programs for conditional branching, iteration and recursion	K_2, K_3
CO 3	Write programs using functions and synthesize a complete program using divide and conquer approach	K_4
CO 4	write programs using arrays, pointers and structures	K_3, K_4
CO 5	Write programs to perform input/output operations on files	K_3, K_4

List of Experiment:

S.No.	**Fundamental Experiments**	
1.	WAP that calculate the simple interest and compound interest when principal, rate of interest and time are given.	
2.	WAP that swaps values of two variables using a third variable and without using third variable	
3.	WAP to compute the roots of quadratic equations.	
4.	WAP that accepts the marks of 5 subjects and finds the percentage marks obtained by the student.It also prints grades according to the following criteria: Between 90-100%----------Print 'A' 80-90%----------Print 'B' 60-80%----------Print 'C' Below 60%----------Print 'D'	
5.	WAP to simulate the calculator (Arithmetic operations: +, -, /, *).	
6.	Write a menu driven program that computes the area of geometrical figures such as rectangle, square, circle and triangle.	
7.	WAP to find the factorial of a given number.	
8.	WAP to print the Fibonacci series.	
9.	WAP to check whether the entered number is prime or not.	
10.	WAP to convert the binary number to decimal number and vice versa	
11.	WAP to print allArmstrong numbers from 1 to N.	
	Arrays	
12.	WAP to find the minimum and maximum element of the array.	
13.	WAP to search an element in an array using Linear Search.	
14.	Write programs to sort the elements of the array in ascending order using Bubble Sort technique.	
15.	WAP to compute the multiplication of two matrices.	
	Pointers and Functions	
16.	WAP to swap the values of two numbers using the call by pointer.	
17.	WAP to compute the factorial of the number using the recursive function factorial ().	
18.	WAP to compute the length of the string using the user defined function xstrlen().	
19.	WAP to concatenate two strings using the user defined function xstrcat().	
	Strings and Structures	

20.	WAP to reverse the string. Also check whether the given string is in palindrome or not.	
21.	WAP to create structure of a student having member name, roll number, age, marks. Also, create an array of structure of 50 students and display the detail of all the students having marks more than 70.	
	File Handling	
22.	WAP to copy the contents of one file onto another file.	
23.	WAP to compare the contents of two files and determine whether they are same or not.	
24.	WAP to check whether the given word exist in a file or not. If yes, then find the number of times it occurs.	
	Dynamic Memory Allocation	
25.	WAP to create an array using dynamic memory allocation.	
	Embedded C	
26.	Installation and working with Keil.	
27.	Implement Intruder alarm system.	

B.TECH FIRST YEAR					
Course Code	**ACSBS0203Z**	**L**	**T**	**P**	**Credits**
Course Title	**Data Structures and Algorithms**	**3**	**1**	**0**	**0**

Course Objectives:

The course covers the basic data structures, algorithm, and efficiency of algorithm, introduction to array, stack, Queue, link list and their implementation. The course aims to give understanding of various searching and sorting algorithms and implementation of tree data structure.

Pre-requisites: Basics of C programming &algorithm

Course Contents / Syllabus

UNIT-I	**Basic Terminologies and Introduction to Algorithm & Data Organization**	**8 hours**

Algorithm specification, Recursion, Performance analysis, Asymptotic Notation - The Big-O, Omega and Theta notation, Programming Style, Refinement of Coding - Time-Space Trade Off, Testing, Data Abstraction

UNIT-II	**Linear Data Structure**	**8 hours**

Array, Stack, Queue, Linked-list and its types, Various Representations, Operations & Applications of Linear Data Structures

UNIT-III	**Non-linear Data Structure**	**8 hours**

Trees (Binary Tree, Threaded Binary Tree, Binary Search Tree, B & B+ Tree, AVL Tree, Splay Tree) and Introduction of Graphs (Directed, Undirected), Various Representations, Operations & Applications of Trees

UNIT-IV	**Searching and Sorting on Various Data Structures**	**8 hours**

Sequential Search, Binary Search, Comparison Trees, Breadth First Search, Depth First Search Insertion Sort, Selection Sort, Shell Sort, Divide and Conquer Sort, Merge Sort, Quick Sort, Heapsort, Introduction to Hashing

UNIT-V	**File & Graph**	**8 hours**

File: Organization (Sequential, Direct, Indexed Sequential, Hashed) and various types of accessing schemes.
Graph: Basic Terminologies, Representations, Operations and Applications of Graphs, Graph search and traversal algorithms and complexity analysis.

Course outcome: At the end of course, the student will be able to

CO1	Analyzeand implement arrays, linked lists, stacks, queues to solve complex problems.	K3, K4
CO2	Compare the computational efficiency of the sorting and searching algorithms.	K4
CO3	Assessthe memory representation oftree and perform various operations on these data structure.	K3
CO4	Apply the concept of recursion to solve the real-world problems.	K3
CO5	Develop the algorithms using graph data structures.	K6

Text Books

1. E. Horowitz, S. Sahni, S. A-Freed, Fundamentals of Data Structures, 2008, Universities Press.
2. A. V. Aho, J. E. Hopperoft, J. D. Ullman,Data Structures and Algorithms, 1983, Pearson.

Reference Books

1. Donald E. Knuth, The Art of Computer Programming: Volume 1: Fundamental Algorithms, 1968, Addison-Wesley.
2. Thomas, H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to Algorithms, 2009, 3rd Edition, The MIT Press.
3. Pat Morin,Open Data Structures: An Introduction (Open Paths to Enriched Learning), 2013, 31st Edition, UBC Press.

B.TECH FIRST YEAR			
Course Code	**ACSBS0253Z**	**L T P**	**Credit**
Course Title	**Data Structures and Algorithms Lab**	**0 0 4**	**0**
Suggested List of Experiments			**CO**
1. Program to create and display linear array			CO1
2. Program to insert a data item at any location in a linear array			O1
3. Program to delete a data item from a linear array			CO1
4. Program to implement linear search in an Array			CO1
5. Program to implement binary search in the sortedarray without recursion			CO1, CO4
6. Program to implement binary search in the sortedarray with recursion			CO1, CO4
7. Program to implement bubble sort in a non-recursive way			CO1, CO4
8. Program to implement selection sort in a non-recursive way			CO1, CO4
9. Program to implement insertion sort in a non-recursive way			CO1, CO4
10. Program to implement merge sort in a non-recursive way			CO1, CO4
11. Program to implement merge sort in a recursive way			CO1, CO4
12. Program to implement Queue Using array			CO1, CO3
13. Program to implement Circular Queue Using array			CO1, CO3
14. Program to implement Stack Operation using array			CO1, CO3
15. Program to implement the Single Linked List a. Insertion b. Deletion c. Traversal d. Reversal e. Searching f. Updation g. Sorting h. Merging			CO1
16. Program to implement the doubly Linked List a. Insertion b. Deletion c. Traversal d. Reversal e. Searching f. Updation g. Merging			CO1
17. Program to implement the circularly Single Linked List a. Insertion b. Deletion c. Traversal d. Reversal e. Searching f. Updation			CO1
18. Program to implement Queue Using linked list			CO1, CO3
19. Program to implement Circular Queue Using linked list			CO1, CO3
20. Program to implement Priority Queue Using linked list			CO1, CO3
21. Program to implement Stack Operation using Linked list			CO1, CO3
22. Program to implement Tower of Hanoi			CO2
23. Program implementing Addition of two polynomials via Linked Lists			CO1
24. Program to implement binary tree using linked list a. Insertion b. Deletion c. Traversal d. Searching			CO1, CO5
25. Program to implement binary search tree using linked list			CO1,

a. Insertion b. Deletion c. Traversal d. Searching		CO5
26. Program to implement heap sort in a non-recursive way		CO1, CO4
27. Program to implement BFS algorithm		CO5
28. Program to implement DFS algorithm		CO5
29. Program to implement the minimum cost spanning tree		CO5
30. Program to implement the shortest path algorithm		CO5
Lab Course Outcome: At the end of course, the student will be able to		
CO1	Write programs for solving mathematical problems using array and linked list.	K3
CO2	Implement concept of recursion to solve complex problem.	K3
CO3	Implement various operations of stack and queue data structure.	K3
CO4	Write efficient sorting, searching programs.	K3
CO5	Implement program to solve real world problem using tree and graph data structure.	K3