ANNEXURE -V

OPEN ELECTIVES offered by Department of Information Technology

Open Electives offered by Department of IT

SNo.	Course Name	L-T-P	CR	Prerequisites	Offered to
3rd Year SEM-1					
1	Python Programming	3-1-0	3	Basic knowledge of programming fundamentals	Other branches
3rd Year SEM-2					
1	Database Management Systems	2-1-0	3	Relational Algebra, Set Theory, knowledge in any program language	Other branches
2	Fundamentals of operating systems	2-1-0	3	Knowledge in Computer Organization	Other branches

PYTHON PROGRAMMING

(Common to CIVIL, CHEM, MECH, EEE and ECE)

COURSE CODE:	**CREDITS 3**
Instruction: 3 Periods & 1Tut/Week	Sessional Marks : 40
End Exam: 3 Hours	End Exam Marks: 60

Prerequisites:

Basic Knowledge of Programming Fundamentals

Course Objectives:

The course should enable the students:

- Describe the core syntax and semantics of Python programming language.
- Illustrate the process of structuring the data using lists, dictionaries, tuples, strings and sets.
- Discover the need for working with the functions, modules and packages.
- Infer the Object-oriented Programming concepts in Python.
- Familiarize the advanced concepts like Iterators, generators, decorators and Indicate the use of regular expressions and built-in functions to navigate the file system.

Course Outcomes:

By the end of the course, the student will be able to:	
1.	Interpret the fundamental Python syntax and semantics and able to solve, test and debug python programs
2.	Fluency in the use of Python control flow statements and Determine the methods to create and manipulate Python programs by utilizing the data structures like lists,
3.	Express proficiency in the handling of functions, modules and packages.
4.	Articulate the Object-Oriented Programming concepts such as encapsulation, inheritance and polymorphism as used in Python.
5.	List the usage and application of iterators, generator, decorators and Identify the commonly used operations involving file systems and regular expressions.

Mapping of Course Outcomes with Program Outcomes:

S. No	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2
CO 1														
CO 2														
CO 3														
CO 4														
CO5														

SYLLABUS

UNIT-I: **10 periods**

Introduction: Installation, Keywords and Identifiers, Statement, Indentation, Comments, Variables, Constants, Literals, Data Types, Type Conversion, I/O, Import, Operators (Arithmetic operators, Comparison operators, Logical operators, Bitwise operators, Assignment operators, Identity operators, Membership operators), Namespace and Scope.

Learning Outcome: At the end of this Unit the student will be able to

- Analyse fundamental advantages of python over the other programming languages.
- Solve, test and debug basic problems using python script.

UNIT-II: **14 periods**

Flow control & Collections: If, If...else, if...elif...else, Nested if, for loop, while loop, Break, Continue and Pass. Numbers, Decimal, Fractions, Mathematics, List, Tuple, String, Set and Dictionary. Data types manipulations (create, Index, Negative indexing, Slicing, change or add elements, delete or remove elements, Methods, Comprehension, Membership Test, Iteration, Operations and Built in Functions)

Learning Outcome: At the end of this Unit the student will beable to

- Implement Flow control statements required real world problems.
- Manipulate python programs by using the python data structures like lists, dictionaries, tuples, strings and sets.

UNIT-III: **12 periods**

Functions: Function, Function argument, Recursion, Anonymous / Lambda functions, Global, Local and Nonlocal variables, Global keyword, Modules and Packages.

Learning Outcome: At the end of this Unit the student will be able to

- Resolve real world problems using python functions.
- Familiarize the usage of Modules and packages to enhance the problem solving.

UNIT-IV: **12 periods**

Object oriented programming: Introduction to OOPs, Class, Object, Constructors, Methods, Inheritance, Method Overriding, Multiple Inheritance, Operator overloading, Encapsulation and Polymorphism.

Learning Outcome: At the end of this Unit the student will be able to

- Design object-oriented programs with Python classes.
- Usage of inheritance, encapsulation, inheritance and polymorphism for reusability.

UNIT-V: **12 periods**

Advanced topics: Iterators, Building Your Own Iterator, Infinite Iterators, Generators, Generator Expression, Closure Function, Decorators, @property decorator, Getters and Setters, RegEx, Match object, datetime, Files(Open, Read, Write, Close) and File Methods,

Learning Outcome: At the end of this Unit the student will be able to

- Interpret the advantages of advanced concepts like iterators, generator, decorators and regular expressions.
- Identify the commonly used operation involved in files for I/O processing.

Text Books:

1. Core Python programming, by W.Chun, Pearson
2. Python Programming : A Modern Approach by Vamsi Kurama, Pearson

Reference Books:

1. How To Think Like A Computer Scientist, Learning With Python, by Allen Downey, Jeffrey Elnker and Chris Meyers
2. Introduction to Python Programming, Gowrishankar S, Veena A, CRC Press/Taylor & Francis.
3. A Beginners Guide to Python 3 Programming by John Hunt, Springer

CASE STUDIES

1. Jack and his three friends have decided to go for a trip by sharing the expenses of the fuel equally. Write a Python program to calculate the amount (in Rs) each of them need to put in for the complete (both to and fro) journey.

The program should also display True, if the amount to be paid by each person is divisible by 5, otherwise it should display False. (Hint: Use the relational operators in print statement.) Assume that mileage of the vehicle, amount per litre of fuel and distance for one way are given.

Test your code by using the given sample inputs.

Verify your code by using the 2nd sample input(highlighted) given below:

Sample Input			Expected
Mileage of the vehicle (km/litre of fuel)	Amount per litre of fuel (Rs)	Distance for one way (kms)	
12	65	96	260.0 True
12	40	190	

2. A three digit number is said to be an "Armstrong number" if the sum of the third power of its individual digits is equal to the number itself.

Example: 371 is an Armstrong number as 371 = 33 + 73 + 13

407 is an Armstrong number as 407 = 43 + 03 + 73

Write a pseudo-code to check whether a given three digit number is an Armstrong number.

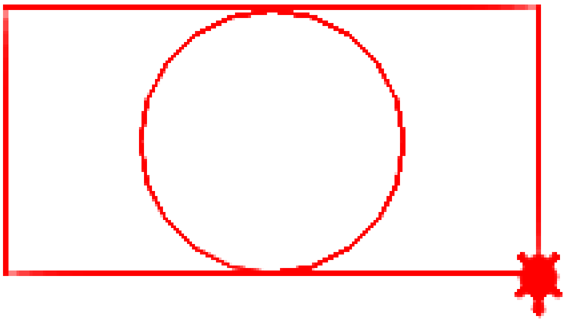
3. A University offering degree courses to students has decided to provide scholarship based on the following details:

Branch of study	Score (%)	Scholarship %	Remarks
Arts	Score is at least 90	50	The student is eligible only for one scholarship% even if both the score conditions are valid for the given branch of study. In such cases, students are eligible for the highest scholarship% applicable among the two.
Arts	Score is an odd number	5	
Engineering	Score is more than 85	50	
Engineering	Score is divisible by 7	5	

If there are 500 students who have joined the university, write a pseudo-code to calculate and display the final fees to be paid by each student. You may accept the branch of study, score and course fee as inputs for each student and calculate the final fees to be paid by each student based on formulae given below:

Scholarship amount=course fee * (scholarship%) Final fee= course fee - scholarship amount

4. Write a program to create the following pattern:



5. Write a python program to find and display the product of three positive integer values based on the rule mentioned below: It should display the product of the three values except when one of the integer value is 7. In that case, 7 should not be included in the product and the values to its left also should not be included. If there is only one value to be considered, display that value itself. If no values can be included in the product, display -1.
Note: Assume that if 7 is one of the positive integer values, then it will occur only once. Refer the sample I/O given below.

Sample Input	Expected Output
1, 5, 3	15
3, 7, 8	8
7, 4, 3	12
1, 5, 7	-1

6. A traveller on a visit to India is in need of some Indian Rupees (INR) but he has money belonging to another currency. He wants to know how much money he should provide in the currency he has, to get the specified amount in INR.
Write a python program to implement a currency calculator which accepts the amount needed in INR and the name of the currency which the traveller has. The program should identify and display the amount the traveller should provide in the currency he has, t o get the specified amount in INR.
Note: Use the forex information provided in the table below for the calculation. Consider that only the currency names mentioned in the table are valid. For any invalid currency name, display -1
.

Currency	Equivalent of 1.00 INR
Euro	0.01417
British Pound	0.0100
Australian Dollar	0.02140
Canadian Dollar	0.02027

7. Write a python program to generate and display the next date of a given date. Assume that date is provided as day, month and year as shown in below table. The input provided is always valid. Output should be day-month-year. Hint: print (day,"-",month,"-",year) will display day-month-year

	Sample Input	Expected Output
Day	1	2-9-2020
Month	9	
Year	2020	

8. Write a python program which finds the maximum number from num1 to num2 (num2 inclusive) based on the followingrules.

1. Always num1 should be less than num2
2. Consider each number from num1 to num2 (num2 inclusive). Populate the number into a list, if the below conditions are satisfied
   a. Sum of the digits of the number is a multiple of 3 b. Number has only two digits
   c. Number is a multiple of 5
3. Display the maximum element from the list

In case of any invalid data or if the list is empty, display -1.

9. Given a string containing uppercase characters (A -Z), compress the string using Run Length encoding. Repetition of character has to be replaced by storing the length of that run. Write a python function which performs the run length encod ing for a given String and returns the run length encoded String.
Provide different String values and test your program.

Sample Input	**Expected Output**
AAAABBBBCCCCCCCC	4A4B8C
AABCCA	2A1B2C1A

10. A hospital wants to know the medical speciality visited by the maximum number of patients. Assume that the patient id of the patient along with the medical speciality visited by the patient is stored in a list. The details of the medical specialities are stored in a dictionary as follows:

{
"P":"Pediatrics
",
"O":"Orthope
dics",
"E":"ENT
}

Write a function to find the medical speciality visited by the maximum number of patients and return the name of the speciality.

Note:

1. Assume that there is always only one medical speciality which is visited by maximum number of patients.
2. Perform case sensitive string comparison wherever necessary.

Sample Input	Expected Output
[101,P,102,O,302,P,305,P]	Pediatrics
[101,O,102,O,302,P,305,E,401,O,656,O]	Orthopedics
[101,O,102,E,302,P,305,P,401,E,656,O,987,E]	ENT

11. Write a python program to display all the common characters between two strings. Return -1 if there are no matching characters.
Note: Ignore blank spaces if there are any. Perform case sensitive string comparison wherever necessary.

Sample Input	Expected output
"I like Python" "Java is a very popular language"	lieyon

12. A teacher is in the process of generating few reports based on the marks scored by the students of her class in a project based assessment.
Assume that the marks of her 10 students are available in a tuple. The marks are out of 25.
Write a python program to implement the following functions:

1. **find_more_than_average():** Find and return the percentage of students who have scored more than the average mark of the class.
2. **generate_frequency():** Find how many students have scored the same marks. For example, how many have scored 0, how many have scored 1, how many have scored 3….how many have scored 25. The result should be populated in a list and returned.
3. **sort_marks():** Sort the marks in the increasing order from 0 to 25. The sorted values should be populated in a list and returned.

Sample Input	Expected Output
list_of_marks = (12,18,25,24,2,5,18,20,20,21)	70.0 [0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2, 0, 2, 1, 0, 0, 1, 1] [2, 5, 12, 18, 18, 20, 20, 21, 24, 25]

13. Write a python function, **check_double(number**) which accepts a whole number and returns
True if it satisfies the given conditions.

1. The number and its double should have exactly the same number of digits.
2. Both the numbers should have the same digits ,but in different order.

Otherwise it should return False.
Example: If the number is 125874 and its double, 251748, contain exactly the same digits, but in a different order.

14. Given a number n, write a program to find the sum of the largest prime factors of each of nine consecutive numbers starting from n.
g(n) = f(n) + f(n+1) + f(n+2) + f(n+3) + f(n+4) + f(n+5) + f(n+6) + f(n+7) + f(n+8)
where, g(n) is the sum and f(n) is the largest prime factor of n

For example, g(10)=f(10)+f(11)+f(12)+f(13)+f(14)+f(15)+f(16)+f(17)+f(18)
=5 + 11 + 3 + 13 + 7 + 5 + 2 + 17 + 3
=66

15. Write a python function, **nearest_palindrome()** which accepts a number and returns the nearest palindrome greater than the given number.

Sample Input	**Expected Output**
12300	12321
12331	12421

16. Assume that a poem is given. Write the regular expressions for the following:

1. Print how many times the letter 'v' appears in the poem.
2. Remove all the newlines from the poem and print the poem in a single line.
3. If a word has 'ch' or 'co', replace it with 'Ch' or 'Co'.
4. If the pattern has characters 'ai' or 'hi', replace the next three characters with **.

Test your code by using the given sample inputs.
Verify your code by using the 2nd sample input(highlighted) given below:

Sample Input	**Expected Output**
If I can stop one heart from breaking, I shall not live in vain; If I can ease one life the aching, Or cool one pain, Or help one fainting robin Unto his nest again, I shall not live in vain.	4 If I can stop one heart from breaking, I shall not live in vain; If I can ease one life the aching, Or cool one pain, Or help one fainting robin Unto his nest again, I shall not live in vain. If I can stop one heart from breaking, I shall not live in vain; If I can ease one life the aChing, Or Cool one pain, Or help one fainting robin Unto his nest again, I shall not live in vain. If I can stop one heart from breaking, I shall not live in vain; If I can ease one life the achi** Or cool one pain, Or help one fai**ng robin Unto hi**est again, I shall not live in vain.

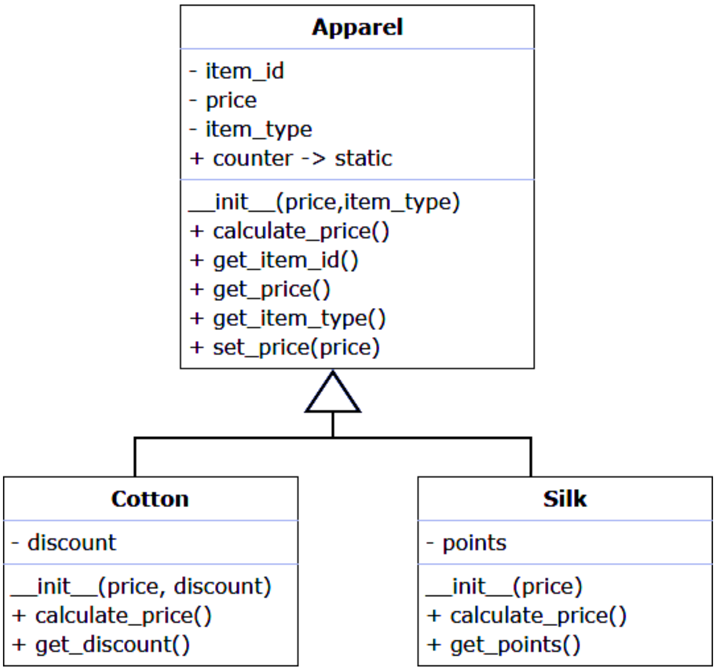
It takes strength for being certain, It takes courage to have doubt. It takes strength for challenging alone, It takes courage to lean on another. It takes strength for loving other souls, It takes courage to be loved. It takes strength for hiding our own pain, It takes courage to help if it is paining for someone.	

17. A university wants to automate their admission process. Students are admitted based on marks scored in a qualifying exam.

A student is identified by student id, age and marks in qualifying exam. Data are valid, if:

- Age is greater than 20
- Marks is between 0 and 100 (both inclusive) A student qualifies for admission, if
- Age and marks are valid and
- Marks is 65 or more Write a python program to represent the students seeking admission in the university.

18. An apparel shop wants to manage the items which it sells. Write a python program to implement the class diagram given below.



19. Write a python program to Find Resolution of JPEG Image

20. Royal Orchid is a florist. They want to be alerted when stock of a flower goes below a particular level.

The flowers are identified using name, price per kg and stock available (in kgs). Write a Python program to implement the above requirement.

DATABASE MANAGEMENT SYSTEMS

COURSE CODE IT324 **CREDITS 3**
Instruction: 2 Periods (& 1Tut) /Week Sessional Marks : 40
End Exam: 3 Hours End Exam Marks: 60

Prerequisite(s): Relational Algebra, Set Theory, knowledge in any program language

Course Objectives

1. Understand basic database concepts, including the structure and operation of the relational data model.
2. Construct simple and moderately advanced database queries using Structured Query Language (SQL).
3. Understand and successfully apply logical database design principles, including E-R diagrams and database normalization.
4. Understand the concept of a database transaction and related database facilities, including concurrency control, locking and protocols.

Course Outcomes

After completion of this course, the students will be able to:
CO-1: Model applications data requirements using conceptual modelling tools like ER diagrams and design database schemas based on the conceptual model.
CO-2: Apply relational database theory and describe relational algebra expression, tuple and domain relation expression for queries.
CO-3: Write SQL commands to create tables and indexes, insert/update/delete data and query data in a relational DBMS. Optimize the database design by applying functional dependency and normalization principles
CO-4:. Examine the serializability of non-serial schedules and compare and contrast the concurrency control protocols.

Mapping of Course Outcomes with POs and PSOs

COs/POs-PSOs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	P010	PO11	PO12	PSO1	PSO2
CO1	3	2	1	1	3				2	2		2	2	1
CO2	3	2	1	1	3				2	2		2	2	1
CO3	3	2	1	1	3				2	2		2	2	1
CO4	3	2	1	1	3				2	2		2	2	1

UNIT-I Introduction **10 Periods**
Introduction to DBMS: Overview, File system vs. DBMS, Structure of DBMS, Levels of Data Abstraction, Database Users and Administrators, E-R model: Entities, Attributes and Entity sets, Relationship and Relationship sets, Features of ER model, Conceptual database design with ER model.
Learning Outcome: At the end of this Unit the student will be able to

- Understand database concepts and structures and query language
- Understand the E R model
- Design ER-models to represent simple database application scenarios

UNIT-II **12 Periods**

Relational model: Integrity constraints over relations and enforcement, Querying relation data, Logical database design, views, destroying/altering tables and views.

Learning Outcome: At the end of this Unit the student will be able to

- Understand the relational model
- Convert the ER-model to relational tables, populate relational database and formulate SQL queries on data
- Explain the basic concepts of relational model, relational database design, relational algebra and relational Calculus

UNIT-III **10 Periods**

SQL: Basic SQL, Query, union, interest, except, Nested Queries, Aggregated Operation, cursors, Database connectivity(ODBC and JDBC), Triggers.

Learning Outcome: At the end of this Unit the student will be able to

- Execute various advance SQL queries
- Write SQL commands to create tables and indexes, insert/update/delete data, and query data in a relational DBMS.
- Perform PL/SQL programming using concept of Cursor Management, Error Handling, Packages and Triggers

UNIT-IV **12 Periods**

Normalization: Introduction to Schema Refinement - Problems Caused By Redundancy, Decomposition, Functional Dependency, Normal Forms (First, Second, Third normal forms, BCNF, Fourth & Fifth normal forms).

Learning Outcome: At the end of this Unit the student will be able

- Understand Functional Dependency and Functional Decomposition.
- Apply various Normalization techniques.
- Improve the database design by normalization.

UNIT-V **10 Periods**

Transaction management: Transaction concept, transactions and schedules, concurrent execution of transactions Concurrency control: Lock management, concurrency control without locking.

Learning Outcome: At the end of this Unit the student will be able to

- understand transactions and their properties (ACID)
- understand the anomalies that occur without ACID
- understand the locking protocols used to ensure Isolation

TEXT BOOKS

1. Raghu Ramakrishnanand Johannes Gehrke, “Database Management Systems”, 3rdEdition, McGraw-Hill, 2003.

REFERENCES

1. Silberschatz, Korth and Sudharshan, “Data Base System Concepts”, 5th Edition, McGraw Hill, 2006.
2. Elmasri, Navathe,“Fundamentals of Database Systems”,5th Edition, Pearson Education, 2007.

FUNDAMENTALS OF OPERATING SYSTEMS

COURSE CODE: **Credits: 3**
Instruction: 2 Periods & 1 Tut/Week Sessional Marks: 40
End Exam: 3 Hours End Exam Marks: 60

Prerequisite: Knowledge in Computer Organization.

Course Objectives:

- □ Understand Functions, Services and structure of Operating Systems. Understand processes, threads, schedulers and explanation of CPU scheduling.
- □ Understand issues related to Process Synchronization and focus on principles of Deadlock and related problems
- □ Comprehend the mechanisms used in Memory Management and Virtual Memory.
- □ Understand the concepts of File System, secondary storage management and Disk Scheduling

Course Outcomes:

1. Analyze basic concepts of operating system and their structures.
2. Analyze various issues related to inter process communication like process scheduling, resource management and deadlocks.
3. Interpret the issues and challenges of memory management.
4. Synthesize the concepts of I/O management, file system implementation and problems related to security and protection

Mapping of course outcomes with program outcomes:

COs/POs-PSOs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	P010	PO11	PO12	PSO1	PSO2
CO1	2	1	2	3	3					1	2	3	3	3
CO2	3	1	2	2				2	2		1	1	1	3
CO3	3	2	2	1	2			3	2		1	2	2	3
CO4	2	2	1	1	2		1	2	1		1	2	2	3

UNIT – I 10 Periods

INTRODUCTION TO OS AND PROCESS MANAGEMENT

Introduction to operating systems, operating system structures, system calls, Process concept, CPU Scheduling: Scheduling criteria, Scheduling algorithms, Multiple processor scheduling, Real time scheduling, Algorithm Evaluation, Operations on processes, Cooperating processes

Learning Outcome: At the end of this Unit the student will be able to

1. Explain Types of operating systems
2. Describe process states and process models
3. Compare processor scheduling algorithm

UNIT – II 10 Periods

PROCESS SYNCHRONIZATION AND DEADLOCK

Process Synchronization: The critical section problem, Synchronization hardware, Semaphores, Classic problems of synchronization, critical regions.

Deadlock: System model, Deadlock characterization, Methods for handling deadlocks. Deadlock prevention.

Learning Outcome: At the end of this Unit the student will be able to

1. Describe race condition & mutual exclusion
2. Identify Deadlocks
3. Apply Deadlock recovery procedure

UNIT – III **10 Periods**

MEMORY MANAGEMENT

Memory Management: Memory Management: Background – Swapping – Contiguous memory allocation – Paging – Segmentation – Segmentation with paging. Virtual Memory: Background– Page replacement algorithms .

Learning Outcome: At the end of this Unit the student will be able to

1. Describe memory management.
2. Differentiate Contiguous and Non contiguous memory.
3. Differentiate physical and virtual primary memory.

UNIT – IV **8 Periods**

FILE SYSTEMS AND ITS IMPLEMENTATION

File System Interface: File concept – Access methods – Directory structure – File system mounting – Protection. File System Implementation : Directory implementation – Allocation methods – Free space management – efficiency and performance – recovery – log structured file systems.

Learning Outcome: At the end of this Unit the student will be able to

1. Apply file management concepts in Operating System
2. Explain Directory structure of Operating System

UNIT – V **10 Periods**

SECONDARY STORAGE STRUCTURES AND PROTECTION

Mass storage structures; Disk structure; Disk attachment; Disk scheduling; Disk management; Swap space management. Protection: Goals of protection, Principles of protection, Domain of protection, Access matrix.

Learning Outcome: At the end of this Unit the student will be able to

1. Describe Disk organization
2. Implement file system security

Text Book:

1. Silberschatz, Galvin, and Gagne, “Operating System Concepts”, Sixth Edition, Wiley IndiaPvt Ltd, 2003.

Reference Books:

1. Andrew S. Tanenbaum, “Modern Operating Systems”, Second Edition, Pearson Education,2004.
2. Gary Nutt, “Operating Systems”, Third Edition, Pearson Education, 2004.
3. Harvey M. Deitel, “Operating Systems”, Third Edition, Pearson Education, 2004.