Savitribai Phule Pune University

SYBBA (CA)
Semester-III (2024 Pattern)

Data Structure & PHP Lab Book

Student Name: _____

College Name: _____

Roll No.: _____ **Division:** _____ **Seat No:** _____

Academic Year: _____

CERTIFICATE

This is to certify that Mr./Ms._____

Seat Number _____ of SYBBA (CA) 2024 pattern Sem-III has successfully completed Laboratory course (Data structure and PHP) in the year _____.

He/She has scored __marks out of 10 (For Lab Book).

Subject Teacher **H.O.D./Coordinator**

Internal Examiner **External Examiner**

Section-I: Data Structure

Section-II: PHP

Introduction

1. About the work book:

This workbook is intended to be used by SYBBA (CA) Semester-III students for Data structure, and PHP Practical assignments. This workbook is designed by considering all the practical topics mentioned in the syllabus.

2. The objectives of this workbook are:

- Defining the scope of the course.
- To bring uniformity in the practical conduction and implementation in all colleges affiliated to SPPU.
- To have continuous assessment of the course and students.
- Providing ready reference for the students during practical implementation.
- Provide more options to students so that they can have good practice before facing the examination.
- Catering to the demand of slow and fast learners and accordingly providing the practice assignments to them.

3. How to use this workbook:

The workbook is divided into Two sections. Section-I is related to Data Structure assignments, Section-II is related to PHP assignments,
Section-I: Data Structure is divided into nine assignments.
Section-II: PHP is divided into assignments.
Students have to perform practical assignments of both subjects from Section-I and Section-II.
Each assignment of all sections has three SETs-A, B and C. It is mandatory for students to complete SET A and SET B in the lab. It also includes practice programs which are expected to be solved by students as home assignments and to be evaluated by subject teachers.

4. Instructions to the students

Please read the following instructions carefully and follow them during Practicals.

- Students are expected to carry this workbook every time they come to the lab for computer practical.
- Students should prepare for the assignment by reading the relevant material which is

mentioned in ready reference and the concepts taught in class.

- The instructor will specify which problems to solve in the lab during the allotted slot and students should complete them and get verified by the instructor. However, students should spend additional hours in the lab and at home to cover all workbook assignments if needed.
- Students will be assessed for each exercise on a scale from 0 to 5.

Not done	0
Incomplete	1
Late Complete	2
Needs improvement	3
Complete	4
Well Done	5

5. Instruction to the Instructors

Make sure that students should follow above instructions.

- Explain the assignment and related concepts in around ten minutes using a whiteboard if required or by demonstrating the software.
- Evaluate each assignment carried out by a student on a scale of 5 as specified above by ticking the appropriate box.
- The value should also be entered on the assignment completion page of the respective Lab course.

6. Instructions to the Lab administrator

You have to ensure appropriate hardware and software is made available to each student. The operating system and software requirements on server side and also client side are as given below:

- Operating System - Windows
- Turbo C/Any C Compiler
- Java script
- WampServer

Assignment Completion Sheet

Student Name:----------

Roll NO:----------

Section-I: Data Structure			
Sr. No.	**Assignment Name**	**Marks (out of 5)**	**Teacher's Sign**
1	Array		
2	Sorting Techniques (Non Recursive)		
3	Sorting Techniques (Recursive)		
4	Searching Techniques		
5	Linked List		
6	Stack		
7	Queue		
8	Trees		
9	Graph		
Total (Out of 45)			
Total (Out of 5)			

Instructor Signature:

Student Name:----------

Roll NO:----------

Section-II: PHP			
Sr. No.	**Assignment Name**	**Marks (out of 5)**	**Teacher's Sign**
1	Basics in PHP		
2	Control Structures and Loops		
3	Arrays and Strings		
4	Functions, Class, and Object		
5	Working With Form and form element		
6	Session and Cookies		
7	Working with the Database		
Total (Out of 35)			
Total (Out of 5)			

Instructor Signature:

Section-I

Data Structure

Assignment No 1: Array (One Dimensional Array)

ARRAY

- An array is a finite ordered collection of homogeneous data elements which provide random access to the elements.
  Finite: - There are specific no. of elements in the array.
  Ordered: - The elements are arranged one by one i.e. first then second and so on.
  Homogeneous: - All the elements are of the same type.

WHAT IS POYNOMIAL

- A polynomial p(x) is the expression in variable x which is in the form ($ax^n + bx^{n-1} + \ldots. + jx + k$), where a, b, c …., k fall in the category of real numbers and 'n' is non negative integer, which is called the degree of polynomial.
- **An essential characteristic of the polynomial is that each term in the polynomial expression consists of two parts:**
  - one is the coefficient
  - other is the exponent

EXAMPLE:

- $10x^2 + 26x$, here 10 and 26 are coefficients and 2, 1 is its exponential value.

Practice Program:

1) Write a menu driven C program to perform the following operation on an integer array:
   a) Display the sum of elements at even subscript position of array
   b) Display the sum of elements at odd subscript position of array
2) Write a C Program to find the largest pair sum in an unsorted array.(hint: find 2 maximum elements from array and then find the sum of both numbers.)
3) Write a C Program to calculate Median of two sorted arrays of different sizes.

SET A:

1) Write a C Program to Count number of occurrences (or frequency) in a given sorted array

   Input: arr[] = {1, 1, 2, 2, 2, 2, 3,}, x = 2
   Output: 4 // x (or 2) occurs 4 times in arr[]
2) Write a C program to accept n elements, store those elements in array and store the square of these numbers in another array and display both the array.
3) Write a C program to Copy one array into another array.

SET B:

1) Write a C program accept the polynomial and display it in format e.g. 6x^4+2x^2+5x^1+3
2) Write a 'C' program to accept n elements store those elements in an array and find and replace a given number.
3) Write a 'C' program to accept two polynomials and find the addition of accepted polynomials.

SET C:

1) Write a ‘C’ program to accept two polynomials and find the Multiplication of accepted polynomials.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 2: Sorting Techniques (Non Recursive)

SORTING

- Sorting means arranging a set of data in some given order or ordering a list of items.

  'Or'

  Sorting is a process of ordering a list of elements in either ascending or descending order.
- List is a collection of record each contains one or more fields. The field which contains unique value for each record is called key field.
- Definition:-

  Sorting is the operation of arranging the records of a table according to the key value of each record

  e.g. consider a telephone directory which consists of 4 field phone number, name, address, pin code .

  So a large data is maintained in the form of records. If we want to search a phone no and name it should be alphabetically sorted then we can search easily. It would be very difficult if records were unsorted.
- The sorting algorithm are divided into two categories

1) Internal sorting-

   Sorting is done on data which is sorted in main memory.
2) External sorting –

   Sorting is done on data which is stored on an auxiliary storage device.

   e.g. hard disk, floppy, tape etc.

- **BUBBLE SORT**
  - This is one of the simplest and most popular sorting methods. The basic idea is to pass through the file sequentially several times.
  - In each pass we compare successive pairs of elements(x[i] with x[i+1]) and interchange the two if they are not in the required order.
  - One element is placed in its correct position in each pass.
  - In the first pass, the largest element will sink to the bottom, second largest in the second pass and so on. Thus a total of n-1 passes are required to sort n keys
  - **Time Complexity**: Base Case: O(n), Worst Case: O(n2), Average Case: O(n2)

Algorithm for Bubble sort:

Step1: Start
Step2: Accept 'n' numbers in array 'A'
Step3: set i=0
Step4: set j=0
Step5: if j<n-i-1 then go to next step else go to step 8
Step 6: if i A[j+1] then interchange A[j] and A[j+1]
Step7: j=j+1 and goto step 5
Step8: i=i+1 and goto step 4
Step9:Stop

- **INSERTION SORT**
  - Insertion sort inserts each item into its proper place in the final list
  - In this the first iteration starts with comparison of 1^{st} element with 0^{th}
  - In the second iteration the 2nd element is compared with the 0^{th} and 1^{st} element and so on.
  - In every iteration an element is compared with all elements
  - The basic idea of this method is to place an unsorted element into its correct position in a growing sorted list of data. We select one element from the unsorted data at a time and insert it into its correct position in the sorted set.
  - E.g. in order to arrange playing cards we pick one card at a time and insert this card hold in the hand.
  - **Time Complexity**: Base Case: O(n) Worst Case: O(n2) Average Case: O(n2)

Algorithm for Insertion Sort:
Step1: Start
Step2: Accept 'n' numbers and store all in array 'A'
Step3: set i=1
Step4: if i<=n-1 then goto next step else goto step 10
Step5: set Temp=A[i] and j=i-1
Step6: if Temp < A[j] && j>=0 then goto next step else goto step 9
Step7: set A[j+1]=A[j]
Step8: set j=j-1
Step9: set A[j+1]=Temp
Step10: Stop

- **SELECTION SORT**
  - It is also called pushdown sort.
  - In this the largest or smallest element is selected by placing it repeatedly till it reaches its proper position.
  - The 0^{th} element is compared with all other elements, if the 0^{th} is found to be greater than the compared element then they are interchanged . In this way after first iteration the smallest element is placed at 0^{th} position. The Procedure is repeated for 1^{st} element and so on.
  - It is simple to implement.
  - The main advantage is that data movement is very less
  - It is not stable. It is an in-place sort
  - **Time Complexity**: Base Case: O(n2) Worst Case: O(n2) Average Case: O(n2)

Algorithm for Selection Sort
Step1: Start
Step2: Accept 'n' numbers and store all in array 'A'
Step3: set i=0
Step4: if i <n-1 then goto next step else goto step 11
Step5: set min=i and j=i+1

Step6: if j< n then goto next step else goto step 9
Step7: if A[j]< A[min] then min=j
Step8: set j=j+1 and goto step 7
Step9: if(min not equal to i) then interchange A[i] and A[min]
Step10: i=i+1 and goto step 4
Step11: Stop

Practice Programs:

1) Write a C program to create an integer array with elements {56,23,11,67,12,89,2} and sort the given array using bubble sort.
2) Write a C program to sort a random array of n integers (value of n accepted from user) by using Bubble Sort / Insertion Sort algorithm in ascending order.
3) Write a C program to create a string array with 5 elements which contains words starting with vowels and sort them using Selection sort.

SET A:

1) Write a C program to accept and sort n elements in ascending order by using bubble sort.
2) Write a C program to accept and sort n elements in ascending order by using insertion sort.
3) Write a 'C' program to accept and sort n elements in ascending order using the Selection sort method.

SET B:

1) Write a C program to create a string array with day of week and sort them using Insertion sort.
2) Write a 'C' program to accept names from the user and sort in alphabetical order using bubble sort.
3) Write a C program to accept and sort n elements in ascending order by using bubble sort and also count the number of swaps. Display the sorted list and total no of swap count.

SET C:

1) Write a C program to read the data from the file "employee.txt" which contains empno and empname and sort the data on names alphabetically (use strcmp) using Bubble Sort.
2) Write a C program to read the data from the file "person.txt" which contains personno and personage and sort the data on age in ascending order using insertion Sort / Selection Sort.
3) Modify the bubble sort, insertion sort and selection sort program of Set A to sort the integers in descending order?

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 3: Sorting Techniques (Recursive)

QUICK SORT

- It is also called “partition exchange sort". The strategy used here is “divide and conquer” i.e we successively partition the list in smaller lists and apply the same procedure to the sub-list. The procedure is as follows:-

Procedure –

- We will consider one element at a time (pivot) and place it in its correct position.
- The pivot is placed in a position such that all elements to the left of the pivot are less than the pivot and all elements to the right are greater.
- The array is partitioned into two parts:- left partition and right partition.
- The same method is applied for each of the partition.
- The process continues till no more partitions can be made. We shall be considering the first element of the partition as the pivot element.

Algorithm:

Step 1: start.
Step 2: A is an array of n elements.
Step 3: lb=0 lb = lower bound
ub = n-1 ub = upper bound.
Step 4: if(lb<ub)
i.e. if the array can be partitioned
j=partition(A,lb,ub) // j is the pivot position

quicksort(A,lb,j-1);
quicksort(A,j+1,ub);

- Now, we must write the function to partition the array. There are many methods to do the partitioning depending upon which element is chosen as the pivot.
- We will be selecting the first element as the pivot element and do the partitioning accordingly.
- We shall choose the first element of the sub- array as the pivot and find its correct position in the sub- array.
- We will be using two variables down and up for moving down and up arrays.

Algorithm for partitioning

Step 1: down=lb+1
Step 2: up=ub
Step 3: pivot=A[lb]
Step 4: perform step 5 to 7 as long as down<up else go to step 8.
Step 5: while (A[down]<=pivot && down<ub) down++;
Step 6: while (A[up]>pivot) up--;
Step 7: if (down<up) interchange A[down] and A[up]
Step 8: interchange A[up] and pivot, j=up, i.e. pivot position=up

Step 9: return up Step 10: stop.

- In this algorithm we want to find the position of pivot i.e. A[lb].
- We use two pointers up and down initialized to the first and last elements respectively.
- We repeatedly increase down as long as the element is < pivot.
- We repeatedly decrease up as long as the element is > pivot.
- If up and down cross each other i.e. up<=down, the correct position of the pivot is up and A[up and pivot are interchanged.
- If up and down do not cross, A[up] and A[down] are interchanged and the process is repeated till they do not cross or coincide.
- Efficiency of quick sort.
  Best case = average case = O(n log n)
  Worst case = $O(n^2)$

MERGE SORT.

- Merging is the process of combining two or more sorted data lists into a third list such that it is also sorted.
- Merge sort follows Divide and Conquer strategy.
  - Divide :- divide an n element sequence into n/2 subsequence.
  - Conquer :- sort the two sequences recursively.
  - Combine :- merge the two sorted sequences into a single sequence.
- In this two lists are compared and the smallest element is stored in the third array.

Algorithm:-

Step 1: start
Step 2: Initially the data is considered as a single array of n elements .
Step 3: divide the array into n/2 sub-array each of length 2^i (I is 0 for 0^{th} iteration). i.e. array is divided into n sub-arrays each of 1 element.
Step 4: merge two consecutive pairs of sub-arrays such that the resulting sub-array is also sorted.
Step 5: The sub-array having no pairs is carried as it is
Step 6: step 3 and 4 are repeated till there is only one sub-array remaining of size
n. Step 7: stop.

Practice Programs:

1) Write a C program to create an integer array with elements {888,111,666,444,222,999,333} and sort the given array using Merge sort.
2) Write a C program to sort a random array of n integers (value of n is accepted from user) by using quick Sort algorithm in ascending order.
3) Write a C program to sort a random array of n integers (value of n accepted from user) by using merge Sort algorithm in ascending order

SET A:

1) Write a C program to accept and sort n elements in ascending order by using merge sort.

2) Write a C program to accept and sort n elements in ascending order by using quick sort.
3) Modify the Quick sort program of SET A to sort the integers in descending order?

SET B:

1) Write a C program to create a string array with months (accept at least 6 months) and sort them using Quick sort.
2) Write a C program to create a string array with at least 5 elements which contains words ending with ‘at’ and ‘an’ sound and sort them using Merge sort.
3) Modify the Merge sort program of Set B to sort the integers in descending order?

SET C:

1) Write a C program to read the data from the file “person.txt” which contains personno, name and personage and sort the data on age in ascending order using merge Sort.
2) Write a C program to read the data from the file “student.txt” which contains roll no, name and age and sort the data on age in ascending order using quick Sort.
3) Read the data from the file student.txt and sort on names in alphabetical order (use strcmp) using Merge sort / Quick sort. Write the sorted data to another file 'sortstudentname.txt'.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 4: Searching Techniques

Searching is the process of finding a value in a list of values. The commonly used searching methods used are linear search and Binary search.

LINEAR SEARCH

- In Linear search, we search an element or value in a given array by traversing the array from the starting, till the desired element or value is found.
- **Time Complexity**: Base Case: O(1) Worst Case: O(n) Average Case: O(n)

Algorithm for Linear Search:

Step 1: Start
Step 2: Accept n numbers in an array num and a number to be searched
Step 3: set i=0 and flag=0
Step 4: if i< n the goto next step else goto
Step 5: Compare num[i] and number If equal then set flag=1 and goto step 7
Step 6: i=i+1 and goto step 4
Step 7: if (flag=1) then Print "Required number is found at location i+1" else Print "Require data not found"
Step 8: Stop

BINARY SEARCH

- Binary Search is used with sorted array or list. So a necessary condition for Binary search to work is that the list/array should be sorted. It works by repeatedly dividing in half the portion of the list that could contain the item.
- **Time Complexity**: Base Case: O(1) Worst Case: O(log n) Average Case: O(log n)

Algorithm for Binary search:

Step 1: Start
Step 2: Accept n numbers in an array num and a number to be searched
Step 3: set low=0, high=n-1 and flag=0
Step 4: if low <= high then middle=(low+high)/2 else goto step 7.
Step 5: if (num[middle]=number) position=middle, flag=1 goto step 7. else if (number< num[middle]) high=middle-1 else low=middle+1
Step 6: goto step 4
Step 7: if flag=1 Print "Required number is found at location position+1" Else Print "Required number is not found.
Step 8: Stop

Practice Programs:

1) Write a C program to linearly search an element in a given array. (Use Recursion).
2) Read the data from file 'employee.txt' containing names of n employees, their qualification and salary. Accept a name of the employee from the user and by using linear search algorithm check whether the name of employee is present in the file or not if present display salary of that employee, otherwise display "Employee not found".

3) Read the data from file ‘player.txt’ containing names of n Player, their game_played and age. Accept a name of the player from the user and by using binary search algorithm check whether the name of player is present in the file or not if present display game_played and age of that player, otherwise display “player not found”.

SET A:

1) Write a C program to accept n elements from user store it in an array. Accept a value from the user and use linear/Sequential search method to check whether the value is present in the array or not. Display proper message.
2) Write a C program to accept n elements from users and store it in an array. Accept a value from the user and use binary search method to check whether the value is present in the array or not. Display proper message. (Students should accept sorted arrays and use a Recursive function).
3) Write a ‘C’ program to create a random array of n integers. Accept a value of n from the user and use Binary search algorithm to check whether the number is present in the array or not. (Students should accept a sorted array and use a Non-Recursive function and also use a random function).

SET B:

1) Write a ‘C’ program to accept the names of cities and store them in an array. Accept the city name from the user and use a linear search algorithm to check whether the city is present in the array or not.
2) Write a C program to accept n elements from users and store it in an array. Accept a value from the user and use a recursive binary search method to check whether the value is present in the array or not. Display proper message. (use any sorting method to sort the array)
3) Read the data from file ‘sortedcities.txt’ containing sorted names of n cities and their STD codes. Accept a name of the city from the user and use a linear search algorithm to check whether the name is present in the file and output the STD code, otherwise output “city not in the list”.

SET C:

1) Write a C program to read the data from file 'cities.txt' containing names of 10 cities and their STD codes. Accept a name of the city from the user and use Binary search algorithm to check whether the name is present in the file and output the STD code, otherwise output “city not in the list”.
2) Write a C program to read the data from file 'student.txt' containing names of 10 students and their roll no. Accept a name of the student from the user and use Binary search algorithm to check whether the name is present in the file and output the roll no, otherwise output “Student name not in the list”.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 5: Linked List

- **Linked list:-**

A linked list is an ordered collection of items which is dynamic in nature i.e. its size varies and each item is 'linked" or connected to another item. It is a linear collection of data elements called nodes.

- **LINKED LIST IMPLEMENTATION-:**

A linked list may be implemented in two ways:

1) Static representation
2) Dynamic representation.

1) Static representation:-

An array is used to store the elements of the list. The elements may not be stored in a sequential order. The correct order can be stored in another array called "link"
The values in this array are pointers to elements in the disk array.

Data array Link array

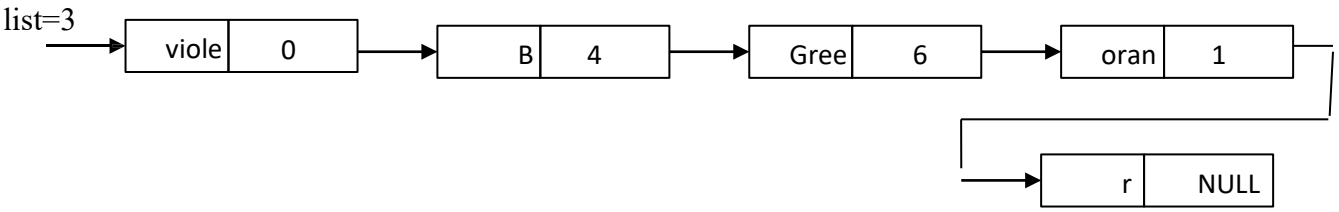
0	Blue
1	Red
2	
3	Violet
4	Green
5	
6	Orange

0	4
1	-1
2	
3	0
4	6
5	
6	1

Data[3] = violet Link[3] = 0
Data[0] = Blue Link[0] = 4
Data[4] = Green Link[4] = 6
Data[6] = Orange Link[6] = 1
Data[1] = Red Link[1] = -1 list end

2) **Dynamic Representation:-**

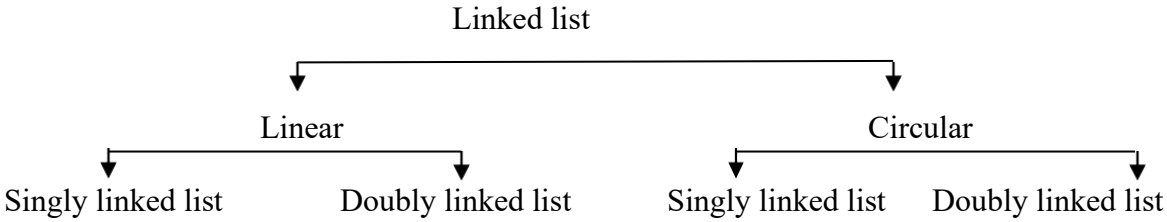
- The static representation uses arrays which is a static data structure and has its own limitations.
- A linked list is a dynamic data structure i.e. the size should increase and when elements are deleted, its size should decrease.
- This cannot be possible using an array which uses static memory allocation i.e. memory is allocated during compile time. Hence we have to use "dynamic memory allocation" where memory can be allocated and de-allocated during run-time.
- Another way of storing a list in memory is by dynamically allocating memory for each node and linking them by means of pointers since each node will be at random memory location. We will need a pointer to store the address of the first node.



List is an external pointer which stores the address of the first node of the list.

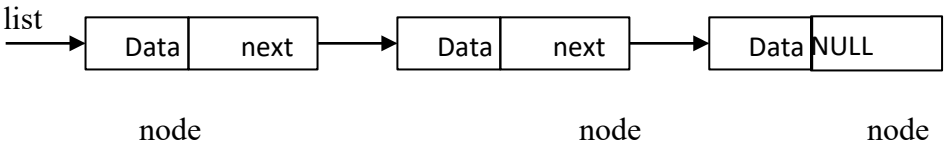
- **TYPES OF LINKED LIST**

1) Singly Linked list
2) Circular linked list
3) Doubly linked list
4) Circular doubly linked list



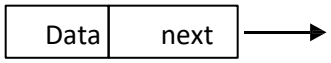
1) Linear linked list

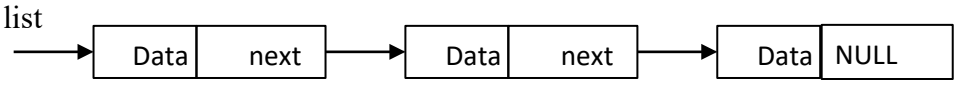
In this list the elements are organized in a linear fashion and list terminates at some point i.e. the last node contains a NULL pointer.



- **Singly linked list-**

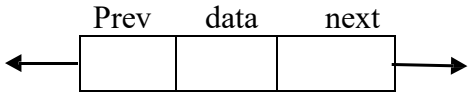
  Each node in this list contains only one pointer which points to the next node.

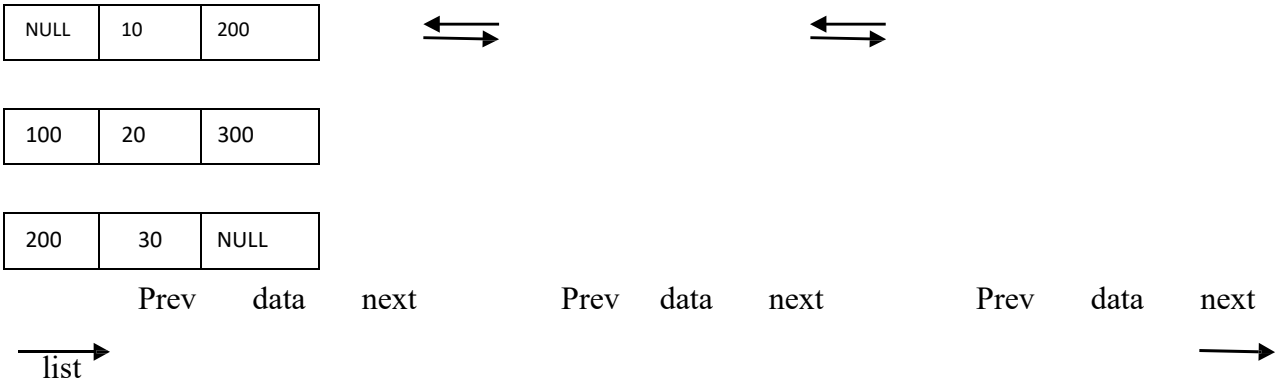




- **Doubly linked list**

  Each node in this contains two pointers, one pointing to the previous node and the other pointing to the next node. This list is used when traversing in both directions is required.

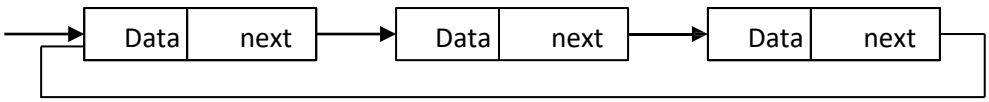


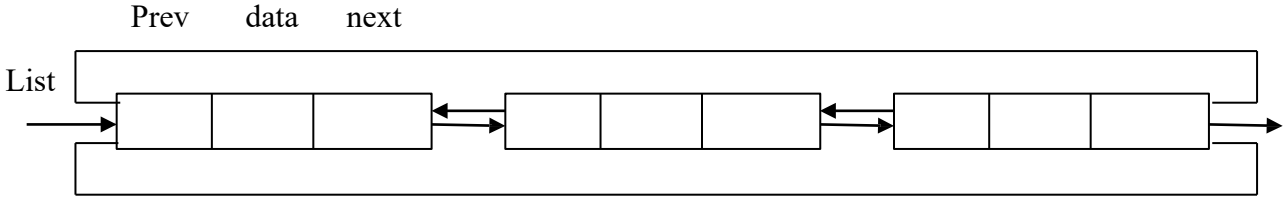


2) Circular list

In this list, the last node does not contain a NULL pointer but points back to the first node i.e. it contains the address of the first node. Each of these lists can be either a singly linked or a doubly list.

Circular singly linked list





Circular doubly linked list

- **OPERATIONS ON A LIST**

The following are some of the basic list operations:-

1) Traversing a list:-
   Visiting each node of the list is called traversal
2) Insertion:-
   A node can be inserted at the beginning, end or in between two nodes of the list.
3) Deletion:-
   Deletion from a list may be done either position-wise or element-wise
4) Display:-
   Display each element of the list.
5) Searching:-
   This process searches for a specific element in the list.
6) Reversing or inversion:-
   This process reverses the order of nodes in the list
7) Concatenation:-
   This process appends the nodes of the second list at the end of the first list i.e. it joins two lists.
8) Computation of length:-
   Count the total no. of nodes in the list
9) Creating a linked list
10) Intersection, union, difference.

Practice Programs:

1) Write a C Program to find largest element of doubly linked list.
2) Write a C Program to interchange the two adjacent nodes in given circular linked list.
3) Write C Program to find length of linked list without using recursion.
4) Write C Program to print alternative nodes in linked list using recursion.

SET A:

1) Write a C program to implement a singly linked list with Create and Display operation.
2) Write a C program to implement a Circular Singly linked list with Create and Display operation.
3) Write a C program to implement a doubly linked list with Create and Display operation.
4) Write a C program to implement a Circular doubly linked list with Create and Display operation

SET B:

1) Implement the following programs by adding the functions one by one in SET A(Question1)
   i) To count total number of nodes and display the count.
   ii) To insert node at the start.
   iii) To reverse the Linked List and display both the list.
2) Write a Menu driven program in C to implement the following functions:
   i) To search the number in the list. If the number is present display the Position of node .If number not present print the message “Number not Found”

ii) To swap mth and nth element of linked list.

iii) To delete node from specific position of linked list.

3) Write a 'C' program to sort elements of a singly linked list in ascending order and display the sorted List.
4) Write a 'C' program to create doubly link list and display nodes having odd value.

SET C:

1) Write a C program to find intersection of two singly linked lists.
2) Write a C program to divide a singly linked list into two almost equal size lists.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 6: Stack

A stack is an ordered collection of items into which items may be inserted and deleted from one end called the top of the stack.
The stack operates in a LIFO (last in first out) manner. i.e. the element which is put in last is the first to come out. That means it is possible to remove elements from the stack in reverse order from the insertion of elements into the stack.
The real life e.g. of the stack are stack of coins, stack of dishes etc. only the topmost plate can be taken and any new plates has to be put at the top.

- **PRIMITIVE OPERATIONS ON A STACK.**

1) Create
2) Push
3) Pop
4) Isempty
5) Isfull
6) Peek

- **STACK IMPLEMENTATION:-**

The stack implementation can be done in two ways:-

1) Static implementation:-

- It can be achieved using arrays. Though it is a very simple method it has few limitations.
- Once a size of an array is declared, its size cannot be modified during program execution.
- The vacant space of stack also occupies memory space.
- In both cases, if we store less arguments than declared, memory is wasted and if we want to store more elements than declared array cannot be expanded. It is suitable only when we exactly know the number of elements to be stored.
- **Operations on static stack:-**

1) Declaring of a stack :-
A stack to be implemented using an array will require.
- An array of a fixed size.
- An integer called top which stored the index or position of the topmost element. We can use a structure for the above purpose.

2) Creating a stack:-
This declaration only specifies the template. The actual stack can be declared as-
STACK s1;

3) initialize a stack:-
When a stack variable is declared the integer top has to be initialized to indicate an empty stack. Since we are using an array the first element will occupy position 0. Hence to indicate an empty stack top has to be initialized to -1

4) Checking whether stack is empty:-
An empty stack can be tested from the value contained in top. If top contains -1 it indicates an empty stack.

5) Checking whether stack is full:-
If the value of top reaches the maximum array index i.e. MAX-1 no more elements can be pushed into the stack.

6) The push operation:-

The element can be pushed into the stack only if it is not full. In such case the top has to be incremented first and the element has to be put in this position.

7) The pop operation:

An element can be removed from the stack if it is not empty. The topmost element can be removed after which top has to decremented

8) The peek operation:

It displays the topmost element of the stack without decrementing the top.

2) Dynamic implementation:-

- Pointers are used for implementation of stack. The linked list is an e.g. of this implementation.
- The limitations noticed in static implementation can be removed using dynamic implementation. The dynamic implementation is achieved using pointers.
- Using pointer implementation at runtime there is no restriction on the no. of elements. The stack may be expandable.
- The memory is efficiently utilized with pointers.
- Memory is allocated only after element is pushed to the stack
- In static representation there is a limitation on the size of the array if less elements are stored, memory will be wasted. To overcome the program the stack can be implemented using linked list.
- In the linked organization
  - The stack can grow to any size.
  - We need not have prior knowledge of the number of elements.
- When an element is popped the memory can be freed. Thus memory is not unnecessary occupied.
- Since random access to any element is not required in a stack, the linked representation is preferred over the sequential organization.

Applications of Stack

1) Inter conversion between infix, postfix and prefix expression.
2) Evaluating the postfix expression.
3) Reversing a string.
4) Reversing each word of the string and many more

There are 3 notations for specifying the operands:-

1) Infix :- if the operator symbols are placed between the operands then the expression is in the infix notation (a+b)*c
2) Postfix :- if the operator symbols are placed after its operands, then the expression is in postfix notation. ab+c*
3) Prefix :- if the operator symbols are placed before its operands, then the expression is in prefix notation. *+abc

Associativity and Precedence rule

Operator	Precedence	Associativity
{},[].()	High	L-R
^ exponent	High	R-L e.g. 3^2^2 =3^4=81
Mul/div *,/	Intermediate	L-R
+,-	Low	L-R

EVALUATE POSTFIX EXPRESSION:

Algorithme

1) Scan the string from Left to right
2) If symbol = = digit then push symbol in the stack.
3) If symbol = = operator then pop 2 elements from stack
   Put first pop element in operand2
   Put second pop element in operand1
   Evaluate the value (operand1 operator operand2) and put the evaluated answer in result
4) Push the result in the stack
5) After all the symbol are finished from symbol column pop the last element from the stack which will be the final result of the postfix expression

INFIX TO POSTFIX CONVERSION

Algorithm

1) Scan the string from left to right
2) If symbol = = opening bracket push in stack
3) If symbol = = closing bracket pop all the elements from stack till we get opening bracket, pop the opening bracket also and then put the pop elements in the postfixstring leaving the opening bracket.
4) If symbol = = alphabet/ digit then put the symbol in postfixstring
5) If symbol = = operator check priority of top element in the stack.
   If priority(top element)>= priority(symbol operator) then pop top element and put it in postfixstring
   If priority(top element)< priority(symbol operator) then push the symbol in the stack
6) Repeat steps 2-5 until the infix expression is scanned.
7) Print the output i.e. postfixstring

INFIX TO PREFIX CONVERSION

Algorithm

1) Scan the string from right to left

2) If symbol = = Closing bracket push in stack
3) If symbol = = opening bracket pop all the elements from stack till we get opening bracket, pop the closing bracket also and then put the pop elements in the Prefixstring leaving the closing bracket.
4) If symbol = = alphabet/ digit then put the symbol in Prefixstring
5) If symbol = = operator check priority of top element in the stack.
   If priority(top element)> priority(symbol operator) then pop top element and put it in Prefixstring
   If priority(top element)<= priority(symbol operator) then push the symbol in the stack
6) Repeat steps 2-5 until the infix expression is scanned.
7) Print the output i.e. print Prefixstring in reverse

STRING REVERSE AND CHECKING PALINDROME STRING:

A string of characters can be reversed by reading each character from a string starting from the first index and pushing it on a stack. Once all the characters have been read, the characters can be popped one at a time from the stack and then stored in another string starting from the first index.

Algorithm to reverse the string:

1) Read the string character by character.
2) Push every character into the stack of characters.
3) When the string becomes empty, pop every character from the stack and attach to the new string.

Algorithm to check palindrome of string:

1) Read the string character by character.
2) Push every character into the stack of characters.
3) When the string becomes empty, pop every character from the stack and attach to the new string.
4) Compare original and reversed string if it matches string is palindrome else it is not palindrome

Practice Programs:

1) Let stack_ptr be a pointer to a stack of integers and item be an integer variable.
   Write functions like Push, Pop, Initialize, Empty, and Full for doing the following tasks. [You may declare additional variables in your functions if needed].
   a. Return the top element of the stack and leave the top element unchanged. If the stack is empty, return INT_MAX.
   b. Return the third element from the top of the stack, provided that the stack contains at least three integers. If not, return INT_MAX. Leave the stack unchanged.
   c. Return the bottom element of stack (or INT_MAX if stack empty), and leave the stack unchanged.

2) Given an expression string exp, write a C program to examine whether the pairs and the orders of "{", "}", "(", ")", "[", "]" are correct in exp.

   Example:
   Input: exp = "[()]{}{[() ()]()}"
   Output: Balanced

Input: exp = "[(])"
Output: Not Balanced

3) Write a C Program to solve Tower Of Hanoi Problem (Use Recursion).
4) Write a C Program to sort a stack using temporary stack.

SET A:

1) Write a C program to implement Static implementation of stack of integers with following operation:
   -Initialize(), push(), pop(), isempty(), isfull(), display()
2) Write a C program to implement Dynamic implementation of stack of integers with following operation:
   -Initialize(), push(), pop(), isempty(), display().
3) Write a C program to reverse each word of the string by using static and dynamic implementation of stack.
   Example: Input - This is an input string
   Output – sihT si na tupni gnirts

SET B :

1) Write a 'C' program which accepts the string and check whether the string is Palindrome or not using stack. (Use Static/Dynamic implementation of Stack).
2) Write a 'C' program to read a postfix expression, evaluate it and display the result. (Use Static/Dynamic implementation of Stack).
3) Write a 'C' program to accept an infix expression, convert it into its equivalent postfix expression and display the result. (Use Static/Dynamic implementation of Stack).

SET C:

1) Write a program to check whether the contents of two stacks are identical.
2) Write a program that copies the contents of one stack into another. The order of two stacks must be identical.(Hint: Use a temporary stack to preserve the order).
3) Write a 'C' program to accept an infix expression, convert it into its equivalent prefix expression and display the result. (Use Static/Dynamic implementation of Stack).

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 7: Queue

A queue is an ordered collection of items from which items may be deleted from (or removed from) one end called the front and into which items may be inserted at the other end called rear.

- **BASIC OPERATIONS ON QUEUE**

1) Create:

   Create a new queue. This operation creates an empty queue.

2) Add or insert:

   Add an element to the queue. A new element can be added to the queue at the rear.

3) Delete:

   Remove an element from the queue. This operation removes the elements, which is at the front of the queue. This operation can only be performed if the queue is not empty.

   The result of an illegal attempt to remove an element from an empty queue is called underflow.

4) isempty:

   Check whether a queue is empty. The operation return true if the queue isempty and false otherwise

5) isfull:

   Check whether a queue is full. The operation return true if the queue isfull and false otherwise

- **REPRESENTATION OF LINEAR QUEUES.**

There are two ways to represent a queue in memory.

1) Static (using an array)
2) Dynamic (using an linked list)

1) Static implementation of queue

   Static implementation or array representation of queue requires three entities-

   - An array to hold queue elements.
   - A variable to hold the index of the front element.
   - A variable to hold the index of the rear element.

   The implementation of a queue using sequential representation is done by using some size MAX and two integer variables front and rear. Initially the front and rear is set to -1. Whenever a new element is added it is added from the rear and whenever an element is to be removed from the front. The queue full condition is when the rear reaches MAX - 1. Queue empty condition is when front is equal to rear.

2) Dynamic implementation of linear queue (using an linked list)

A queue can be considered as a list in which all insertions are made at one end called the rear and all deletions from the other end from front.

A queue can be easily represented using a linked list. The front and rear will be two pointers pointing to the first and last node respectively.

Practice Programs:

Write a C program to Implement Static implementation of circular queue of integers which includes operation as: a) Initialize() b) insert() c) delete() d) isempty() e) isfull() f) display() g) peek()

1) Write a C program to Implement Dynamic implementation of circular queue of integers includes operation as : a)Initialize() b) insert() c)delete() d) isempty() e)display() f) peek()
2) Write a C Program to implement Deque using doubly linked list

SET A:

1) Write a C program to Implement Static implementation of Queue of integers with following operation:
   -Initialize(), insert(), delete(), isempty(), isfull(), display(), peek()

2) Write a program to reverse the elements of a queue (Use Static implementation of Queue)

SET B:

1) Write a C program to Implement Dynamic implementation of Queue of integers with following operation:
   -Initialize(), insert(), delete(), isempty(), display(), peek()

2) Write a program to reverse the elements of a queue (Use Dynamic implementation of Queue)
3) Write a C program to Implement Static implementation of circular queue of integers with following operation:
   -Initialize(), insert(), delete(), isempty(), isfull(), display(), peek()

SET C:

1) Write a c program to simulate waiting list operations of railway reservation system.
2) Implement a priority of integers using a static implementation of the queue and implementing the below two operations. Write a menu driven program
   a) Add an element with its priority into the queue.
   b) Delete an element from the queue according to its priority.
3) A doubly ended queue allows additions and deletions from both the ends that is front and rear. Initially additions from the front will not be possible. To avoid this situation, the array can be treated as if it were circular. Implement a queue library (dstqueue.h) of integers using a static implementation of the circular queue and implementing the nine operations : 1)init(Q), 2) isempty(Q) 3) isFull(Q) 4)getFront(Q), 5)getRear(Q), 6)addFront(Q,x), 7)deleteFront(Q) 8) addRear(Q,x) 9)deleteRear(Q)

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Instructor

Assignment No 8: Tree

Definition of tree:-

A tree is a finite set of one or more nodes such that:- there is a specially designated node called the root. The remaining nodes are partitioned into n>=0 disjoint sets T1……Tn where each of these sets is a tree. T1…..Tn are called sub-trees of the root.

Binary Search Tree (BST) is a tree in which all the nodes follow the below-mentioned properties-

- The value of the key of the left sub-tree is less than the value of its parent (root) node's key.
- The value of the key of the right sub-tree is greater than or equal to the value of its parent (root) node's key.
- The left and right sub tree each must also be a binary search tree.

`Thus, BST divides all its sub-trees into two segments; the left sub-tree and the right sub-tree and can be defined as – left_subtree (keys) < node (key) ≤ right_subtree (keys)`

The operations on binary search tree are

init (T) – creates an empty Binary search tree by initializing T to NULL
insert (T, x) – inserts the value x in the proper position in the Binary search tree **search (T, x)** – searches if the value x is present in the search tree
inOrder (T) – displays the node using inorder traversal of binary search tree
postOrder (T) – displays the node using postorder traversal of binary search tree
preOrder (T) – displays the node using preorder traversal of binary search tree

Practice Programs:

1) Write a C program to find all the ancestors of a given node in a binary tree.
2) Write a C program to implement binary search tree so that it handles duplicate keys properly. That is, if a key is already in the tree then the new value should replace the old rather than adding another node with the same key.
3) Write a C program to create binary search tree of integers and perform following operations using non- recursive functions
   - Preorder traversal
   - Inorder traversal
   - Postorder traversal

SET A:

1) Write C programs to implement create and display operations for binary tree.
2) Write C programs to implement create and display operations for binary search tree.
3) Write a C Program to find the product of all leaf nodes of a binary tree

SET B:

1) Write a C Program to implement the following functions on Binary Search Tree
   - To insert a new element in the tree.
   - To search an element in a tree and give the proper message.
2) Write a C Program to implement the following functions on Binary Search Tree

- To create a mirror image of the tree.
- To count non-leaf nodes.

3) Write a C Program to implement the following functions on Binary Search Tree
   - To count leaf nodes.
   - To count total number of nodes.

SET C:

1) Write C programs to create and display the elements using Inorder traversal.
2) Write a C program to create binary search tree of integers and perform following operations: -
   - Preorder traversal
   - Postorder traversal

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: WellDone [ ]

Signature of Instructor

Assignment No 9: Graph

A graph consists of a set of vertices and a set of edges. The two main ways of representing graphs are adjacency matrix representation and adjacency list representation. In adjacency matrix representation of a Graph with n vertices and e edges, a two dimensional n x n array , say a , is used , with the property that a[i,j] equals 1 if there is an edge from i to j and a[i,j] equals 0 if there is no edge from i to j.
In the adjacency list representation of a graph with n vertices and e edges, there are n linked lists, one list for each vertex in the graph.

The usual operations on graph are:
Indegree(i) – returns the indegree (the number of edges ending on) of the ith vertex
Outdegree(i) – returns the outdegree(the number of edges moving out) of the ith vertex
displayAdjMatrix – displays the adjacency matrix for the graph

Practice Programs:

1) Write a C Program to count the number of edges in an undirected graph.
2) Write a C Program to trace all the paths of a directed graph from the given source node to the destination node .Given the adjacency representation of a directed graph, find all the paths of the graph from source to destination.
3) Write a c program to find whether cycle in present in graph (use Directed graph)

SET A:

1) Write a C program to read a graph as an adjacency matrix and display the adjacency matrix.
2) Write a C program to display the total degree of each vertex.
3) Write a C program to display Indegree and outdgree degree of each vertex.

SET B:

1) Write a C program to convert an adjacency matrix into an adjacency list. Display the adjacency list.
2) Write a C program to traverse graph by using BFS.
3) Write a C program to traverse graph by using DFS.

SET C:

1) Implement a program to read a graph as an adjacency matrix. Find the transpose of the matrix for display accepted adjacency Matrix and Adjacency Matrix and List of transpose of the matrix.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: WellDone [ ]

Signature of Instructor

Section II

PHP

Assignment 1: Basics in PHP

Basics of PHP

For learning PHP, we have to learn the following basic points:

PHP Delimiters:

PHP is an embedded application, for writing PHP code we have to use its delimiters

<? php => starting delimiter and

?> => ending delimiter.

Syntax:

```
<? php

?>
```

Data Types:

PHP supports the following data types:

- String
- Integer
- Float (floating-point numbers - also called double)
- Boolean
- Array
- Object
- NULL
- Resource

Operators in PHP

Types of operators that can be used in PHP programs are

There are the following types of operators:

- Arithmetic operators:- + , - , * , / , % ,**
- Assignment operators :- =
- Comparison operators :- <, >, <=, ==, ===, !=, <>, !==
- Increment/Decrement operators :- ++,--
- Logical operators: - &&, ||, !
- String operators:- .(concatenation)
- Conditional assignment operators: - ?,:

***Note:** Students can design HTML form to accept input from the user as per the requiremet of the program.

Practice Programs:

1. Write a PHP script to perform arithmetic operations on two numbers (Addition, Subtraction, Multiplication Division).
2. Write a PHP script to display a maximum of two numbers using a conditional operator.
3. Write a PHP script that will perform pre and post-increment of a number. (Example ++a, a++).

SET A:

1. Write a PHP Script to display Quotient and Remainder of the division of two variables.
2. Write a PHP Script to swap the values of two variables.
3. Write a PHP Script which will convert temperatures from Celsius(C)to Fahrenheit (F). (Hint: C=5.0/9(F-32)

SET B:

1. Write a PHP Script to display the surface area and volume of a cuboid.
   (Hint: surface area=2(lb+lh+bh), volume = l*b*h)
2. Write a PHP Script to calculate the area of Circle, Square, and Rectangle.
3. Write a PHP Script to display the total and percentage of Marks of Subjects (Out of 100) Data Structure, Digital Marketing, PHP, SE, and Bigdata.

SET C:

1. Write a PHP Script to calculate the total cost of AIR Ticket Reservation and display the details for Name, Address, Contact No, Source, Destination, Date of journey, Gender of passenger, No of Persons, Price per Ticket, etc.

Assignment Evaluation

0: Not Done [ ]	**1: Incomplete [ ]**	**2: Late Complete [ ]**
3: Need Improvement [ ]	**4: Complete [ ]**	**5: Well Done [ ]**

Signature of Instructor

Assignment 2: Control Structures and Loops

Conditional Statements

Conditional statements are used to check conditions and the programmer can accordingly display the results. PHP supports the following conditional statements.

1. if Statement
2. if else Statement
3. elseif Statement
4. switch Statement

Name/Use	Syntax	Example

if Statement:
It is used to check a condition. If the condition is true the corresponding body of if statement is executed.
	if(Condition)
	{
	Statements;
	}
		<?php
		$n=10;
		if ($n%2==0)
		{
		echo “Number Is Even”;
		}
		?>
		Output: Number Is Even

if else Statement:
It checks a condition and if the condition is true the corresponding body of the if statement is executed otherwise else part is executed.
	if(Condition)
	{
	Statements;
	}
	else
	{
	Statements;
	}
		<?php
		$n=10;
		if ($n%2==0)
		{ echo “Number Is Even”;}
		else
		{ echo “Number Is Odd”;}
		?>
		Output: Number Is Even

elseif Statement:
It checks more than one condition.
	if (condition)
	{ Statements;}
	elseif (condition)
	{ Statements;}
	else
	{ Statements;}
		<?php
		$a=10;
		$b=20;
		if ($a==$b)
		{ echo “a and b are same”;}
		elseif ($a<$b)
		{ echo “a is less than b”;}
		else
		{ echo “a is greater than b”;}
		
		?>
		Output:a is less than b

switch Statement It checks the result of the expression with multiple conditions (cases). The case is	switch (expression) { case value1: Statements	<?php $num=2; switch($num) {

executed for which the match is found.	break; case value2: Statements break;..... default: Code to be executed if all cases are not matched; }	Case 1: echo “One”; break; Case 2: echo “two”; break; Case 3: echo “Three” break; Case 4: echo “Four; break; Case 5: echo “Five”; break; default: echo “Invalid Number”; ?> **Output:** Two

Loops

Loops in php are used to execute a similar group of statements. PHP supports the following 4 types of loops.

1. for Loop
2. while Loop
3. do…while Loop
4. foreach Loop

Name/Use	**Syntax**	**Example**

for Loop
It executes a block of code for a specified number of times.
	for (initialization; condition; increment)
	{
	code to be executed;
	}
		<?php
		For($j=1; $j<=5; $j++)
		{
		echo $j;
		}
		?>
		Output: 12345

while Loop
It executes a block of code until the condition specified is true.
	while(expression)
	{
	Statements;
	}
		<?php
		$j=1;
		while($j<=5)
		{
		echo $j;
		$j++;
		}
		?>

		Output: 12345

do…while Loop
It executes ablock of code once and then repeats the loop as long as a special condition is true.
	do
	{
	code to be executed;
	}
	while (condition);
		<?php
		$j=1;
		do{
		echo $j;
		$j++;
		} while($j<=5);
		
		?>
		Output: 12345

foreach Loop
It is used to traverse the array and the block of code is executed for each element of the array.
	foreach (array as value) {
	code to be executed;
	}
		<?php
		$array = array(1, 2, 3, 4, 5);
		foreach($array as $value)
		{
		echo "Value is $value";
		}
		?>
		Output:
		Value is 1Value is 2Value is 3Value is 4Value is 5

***Note: S**tudents can design HTML form to accept input from the user as per the requirement of the program.

Practice Programs:

1. Write a PHP Script to display a maximum of two numbers.
2. Write a PHP Script to check whether a number is positive or negative.
3. Write a PHP Script to display a Multiplication table of a number

SET A:

1. Write a PHP Script to check whether a year is a leap or not.
2. Write a PHP Script which will perform the Addition, Subtraction, Multiplication, and Division of two numbers as per the choice. (Use Switch Case)
3. Write a PHP Script to display the grade of the student according to percentage. Use the following conditions:
   Percentage <40 => Grade=”Fail”
   Percentage >= 40 and Percentage <=50 => Grade= “Pass Class”
   Percentage >=50 and Percentage <=60 => Grade= “Higher Second Class”
   Percentage >60 and Percentage <=70 => Grade= “First Class”
   Percentage >70 => Grade= “First Class with Distinction”

SET B:

1. Write a PHP Script to display prime numbers between 1 to 50.

2. Write a PHP Script to display a perfect numbers between 1 to100.
3. Write a PHP Script to display the reverse of a number. E.g. 607 =>706
4. Write a PHP Script to display Armstrong numbers between 1 to 500.

SET C:

1. Write a PHP script to display a number in words (Use Switch case)
   e.g. 345–three four five
2. Write a PHP script to change the background color of the browser using a switch statement according to the day of the week.
3. Write a PHP script to count the total number of even and odd numbers between 1 to 1000.

Assignment Evaluation

0: Not Done [ ] **1: Incomplete [ ]** **2: Late Complete [ ]**
3: Need Improvement [ ] **4: Complete [ ]** **5: Well Done [ ]**

Signature of Instructor

Assignment 3: Arrays and Strings

Strings in PHP

A string is a sequence of characters. There are two types of strings.

1. **Single-Quoted String:**In this type, characters are enclosed with a single quotation mark(‘’);

Examples:

‘Hello World’
‘Amar’
‘Pune’

The limitation of a single-quoted string is that variables are not interpolated.

Example:

```
<?php
$name=‘Amar’;
$str='Hello $name';
echo $str;
?>
```

Output:

Hello $name

2. **Double-Quoted String:** In this type,characters are enclosed with double quotation marks (“ ”).

PHP interpreter interprets variables and special characters inside double-quotes.

Example:

```
<?php
    $name=‘Amar’;
    $str=“Hello $name”;
    echo $str;
?>
```

Output:

Hello Amar

It expands the many PHP escape sequences. The escape sequences recognized by PHP in double-quoted strings are as follows:

Escape Sequence	Meaning
\n	New Line
\r	carriage return
\t	horizontal tab
\v	vertical tab
\e	escape
\f	form feed
\\	Backslash
\$	dollar sign
\"	double-quote

String Functions

PHP provides approximately one hundred functions for string manipulations. Some of the functions that can be performed on strings are:

- Compare two strings
- Find a String In AnotherString
- Find Out How Many Instances of A String Occur In AnotherString
- Return Part of aString
- Replace Part of aString
- Trim Whitespace From The Ends of aString
- Make An Entire String Lowercase or uppercase

Name	Use	Example
strlen()	It is used get string length.	<?php $input = 'Sunayana'; echo strlen($input); ?> **Output:**8
trim()	It used to remove the whitespaces and other characters.	<?php $input = " Programming in PHP \n"; echo trim($input); ?> **Output:** Programming in PHP
ltrim()	It used to strip whitespace or other characters from the beginning of a string.	<?php $input = " Programming in PHP \n"; echo trim($input); ?> **Output:** Programming in PHP \n
rtrim()	It is used to remove the	<?php

	white spaces from end of the string.	$input = " Programming in PHP \n"; echo trim($input); ?> **Output:** Programming in PHP
strtolower()	It converts the whole string into lower case.	<?php echo strtolower("DYPATIL ACS"); ?> **Output:** dypatil acs
strtoupper()	It converts the whole string into upper case.	<?php echo strtoupper("d y patil acs "); ?> **Output:** D Y PATIL ACS
ucfirst()	It used to convert the first character of a string to upper case.	<?php echo ucfirst("dypatil"); ?> **Output:**Dypatil
ucwords()	It used to convert the first character of a string to upper case in each string	<?php echo ucwords("d y patil pimpri"); ?> **Output:** D Y Patil Pimpri
strcmp()	It is used to compare two strings. If two string are equal it returns 0 otherwise 1.	<?php echo "The result is "; echostrcmp("Hello world!","Hello world!"); ?> Output: The result is 0
substr()	Returns a part of a string	<?php echo substr("D Y Patil",2); ?> **Output:** Y Patil
substr_replace()	It used to replace the part of string with another string	<?php echosubstr_replace("HelloWorld","Good Morning",0); ?> **Output:** Good Morning
substr_compare()	It used to compare two string format with a specific start position	<?php echo substr_compare("Hello","world",0)." "; echo substr_compare("abcde","de",1,3)." "; ?> **Output:** -1 -1
substr_count()	It used to count the number of sub strings	<?php echo substr_count("HelloWorld","World"); ?> **Output:** 1
strrev()	It is used to reverse a	<?php

	string.	echo strrev("sairamkrishna"); ?> **Output:**anhsirkmarias

str_pad()	It pads a string to a new length.	<?php
		//Pad to the right side of the string, to a new length of 20 characters:
		
		$str = "Hello World";
		echo str_pad($str,15,"=");
		?>
		Output: Hello World====

explode()	It is used to split a string by string	<?php
		//Decomposing string
		//Break a string into an array:
		$str = "Hello world. It's a beautiful day.";
		$arr=explode(" ",$str);
		print_r ($arr);
		echo"
		
		";
		$str = "one|two|three|four";
		$arr=explode('\|',$str);
		print_r ($arr);
		?>
		Output:
		Array ([0] => Hello [1] => world. [2] =>It's [3] => a [4] => beautiful [5] => day.)
		
		Array ([0] => one [1] => two [2] => three [3] => four)

implode()	It creates a string from an array of smaller string.	<?php
		$arr = array('Hello','World!','Beautiful','Day!');
		echo implode(" ",$arr);
		echo "
		
		";
		$arr = array('Hello','World!','Beautiful','Day!');
		echo implode(",",$arr);
		?>
		Output:
		HelloWorld! Beautiful Day!
		
		Hello,World!,Beautiful,Day!

strpos()	It is used to find the position of first occurrence of a string inside another string.	<?php echostrpos("I love php, I love php too!","php")." "; ?> **Output:** 7

strstr()	It is used to find the first occurrence of a string and returns from that small string onwards.	<?php echostrstr("Hello world!","world")." "; echostrstr("w3resource.com","."); ?> **Output:** world! .com

Arrays in PHP

An array is a collection of different data elements. Multiple elements can be stored using an array under a single name.

Declaration of an Array

An array can be defined/declared by using **array ()** function.

Syntax:

```
$a=array (10, 20, 30, 40);
$colors = array("Red", "Blue, "Yellow");
```

There are following types of an array:

1. Indexed array.
2. Associative array.
3. Multidimensional array.

An array is organized as an ordered collection of (key,value) pairs. In PHP there are three types of arrays:

a) Indexed array: It is an array with a numeric index starting with 0. There are two ways to create an Indexed array:

Example:

```
$num=array (10, 20);
        OR
$num[0]=10;
$num[1]=20;
```

b) Associative array: Associative arrays are arrays that use named keys that you assign to them. There are two ways to create an associative array:

Example:

```
$age = array("Sagar"=>"35", "Abhijeet"=>"37", "Ishwar"=>"43");
                    OR
```

```
$age['Sagar'] = "35";
$age['Abhijeet'] = "37";
$age['Ishwar'] = "43";
```

c) Multidimensional array: A multidimensional array is an array containing one or more arrays. In this type of array, multiple arrays can be defined in a single array.

Example:

```
$cars = array(
  array("Swift",20,30),
  array("Dezire,40,50),
  array("Mercedez",6,7),
  array("Scoda",12,15)
  );
```

Array Functions

Name	Use	Example
array_chunk()	It is used to split an array into chunks of a given size	<?php $a=array("10","20","30","40"); print_r(array_chunk($a,2); ?> **Output:** Array([0] => Array([0] =>10 [1] =>20) [1] => Array ([0] =>30 [1] =>40))
array_combine ()	It is used to combine two arrays into one, values of the first array are the keys and values of the second array are the values in the combined array.	<? php $X=array("a","b","c"); $Y=array("100","200","300"); $Z=array_combine($X,$Y); print_r($Z); ?> **Output :**Array([a]=>100, [b]=>200, [c]=>300)
array_diff ():	It is used to compare the values of two arrays and return the difference	<?php echo "<font size=14>"; $a=array(1,2,3,4,5); $b=array(4,2,6); $c=array_diff($a,$b); print_r($c); ?> **Output:** Array ([0] => 1 [2] => 3 [4] => 5)
array_intersect()	It returns the common elements of two arrays.	<?php $a=array(1,2,3,4); $b=array(4,5,6,2); $c=array_intersect($a,$b);

		print_r($c); ?> **Output:** Array([1]=>2, [2]=>4);
array_flip()	Exchanges all keys with their associated values in an array.	<?php $a = array("a"=>1, "b"=>2, "c"=>3, "d"=>4, "e"=>5); print_r(array_flip($a)); ?> **Output:** Array ([1] => a [2] => b [3] => c [4] => d [5] => e)

array_splice()	It removes and replaces specified elements of an array	<?php
		$a=array(10,20,30,40,50,60);
		$b=array_splice($a,2,3);
		print_r($a);
		print_r($b);
		?>
		Output: Array ([0] => 10 [1] => 20 [2] => 60)
		Array ([0] => 30 [1] => 40 [2] => 50)

array_slice()	It returns selected parts of an array. It returns the sequence of elements from the array array as specified by
	the offset and length para meters.
		<?php
		$a=array(1,2,3,4,5,6);
		$b=array_slice($a,2,3);
		print_r($a);
		echo "
		";
		print_r($b);
		?>
		Output:
		Array ([0] => 1 [1] => 2 [2] => 3 [3] => 4 [4] => 5 [5] => 6)
		Array ([0] => 3 [1] => 4 [2] => 5)

array_reverse()	Returns an array in the reverse order.	<?php $a=array(1,2,3); $d=array_reverse($a); print_r($d) ?> **Output:** Array ([0] => 3 [1] => 2 [2] => 1)

array_key_exists( )	This function is used to check if an element exists in thearray	<?php
		$a=array("a"=>"ABC","p"=>"PQR","x"=>"XYZ");
		if(array_key_exists("p",$a))
		echo "Key Exists";
		else
		echo "Key Does not Exists";
		?>
		Output: Key Exists

array_push()	This function add the	<?php

	new element at the end of an array.	$a = array("a"=>"banana","b"=>"apple","c"=>"orange");
		print_r(array_push($a, "Straberry"));
		print_r($input);
		?>
		Output: 4 Array ([a] => banana [b] => apple [c] => orange [0] =>Straberry)

array_pop()	This functions remove last element of an array.	<?php $a=array("a"=>"banana","b"=>"apple","c"=>"orange"); print_r(array_pop($a)); ?> **Output:** orange
array_shift()	It removes the first element from an array, and returns the value of the removed element	<?php $a = array("a"=>"banana","b"=>"apple","c"=>"Mango"); print_r(array_shift($a)); ?> **Output:** banana

array_unshift()	It adds one or more elements to the beginning of an array	<?php
		$a = array("orange", "banana");
		array_unshift($a, "apple");
		print_r($a);
		?>
		Output: Array ([0] => apple [1] => orange [2] => banana)

array_sum()	It returns the addition of array elements.	<?php $a=array(1,2,3); $sum=array_sum($a); echo "Sum=$sum"; ?> **Output:** Sum=6
array_product()	It returns the product of array elements.	<?php $a = array(5,6); print_r(array_product($a)); ?> **Output:**30
array_unique()	It removes duplicate values from an array	<?php $a = array("a" => "green", "red", "b" => "green", "blue", "red"); $result = array_unique($a); print_r($result); ?>

		Output: Array ([a] => green [0] => red [1] => blue)
extract()	It creates local variables from an array.	**?**php $a = "Original"; $my_array = array("a" => "Cat","b" => "Dog", "c" => "Horse"); extract($my_array); echo "\$a = $a; \$b = $b; \$c = $c"; ?> **Output:** $a = Cat; $b = Dog; $c = Horse
compact()	Create array containing variables and their values	<?php $city = "Pune"; $state = "Mumbai"; $result = compact("city", "state"); print_r($result); ?> **Output:**Array ([city] => Pune [state] => Mumbai)

in_array()	Checks if a specified value exists in an array	<?php
		$a=array(10,20,30,40,50,60);
		if(in_array(40,$a))
		echo "Element Available in array";
		else
		echo "Element not available in array";
		?>
		Output: Element Available in array

count()	It gives number of elements in an array.	<?php $a=array(10,20,30,40,50,60); echocount($a); ?> **Output:** 6

Array Sorting Functions

Name	Use	Syntax

sort()	It sorts array in ascending order.	<?php
		$a=array("mh","ap","LM","za");
		print_r($a);
		echo "
		";
		sort($a);
		echo "
		 After Sorting
		";
		print_r($a);
		?>

		Output: Array ([0] =>mh [1] =>ap [2] => LM [3] =>za) After Sorting Array ([0] => LM [1] =>ap [2] =>mh [3] =>za)

rsort()	It sorts array in descending order.	<?php
		$a=array("mh","ap","LM","za");
		print_r($a);
		echo "
		";
		rsort($a);
		echo "
		 After Sorting
		";
		print_r($a);
		?>
		Output: Array ([0] =>mh [1] =>ap [2] => LM [3] =>za)
		After Sorting
		Array ([0] =>za [1] =>mh [2] =>ap [3] => LM)

asort()	It sorts associative array in ascending order as per the values	<?php
		$b=
		array("X"=>"XYZ","A"=>"ABC","L"=>"LMN");
		print_r($b);
		asort($b);
		echo "
		 After Sorting
		";
		print_r($b);
		?>
		Output: Array ([X] => XYZ [A] => ABC [L] => LMN)
		After Sorting
		Array ([A] => ABC [L] => LMN [X] => XYZ)

arsort()	It sorts associative array in descending order as per the values.	<?php
		$b=
		array("X"=>"XYZ","A"=>"ABC","L"=>"LMN");
		print_r($b);
		arsort($b);
		echo "
		 After Sorting
		";
		print_r($b);
		?>
		Output:
		Array ([X] => XYZ [A] => ABC [L] => LMN)
		After Sorting
		Array ([A] => ABC [L] => LMN [X] => XYZ)

ksort()	It sorts associative array in ascending order as per the keys.	<?php
		$b=
		array("X"=>"XYZ","A"=>"ABC","L"=>"LMN");
		print_r($b);
		ksort($b);

		echo " After Sorting "; print_r($b); ?> **Output:** Array ([X] => XYZ [A] => ABC [L] => LMN) After Sorting Array ([A] => ABC [L] => LMN [X] => XYZ)

krsort()	It sorts associative array in descending order as per the keys.	<?php
		$b=
		array("X"=>"XYZ","A"=>"ABC","L"=>"LMN");
		print_r($b);
		krsort($b);
		echo "
		 After Sorting
		";
		print_r($b);
		?>
		Output:
		Array ([X] => XYZ [A] => ABC [L] => LMN)
		After Sorting
		Array ([X] => XYZ [L] => LMN [A] => ABC)

***Note: S**tudents can design HTML form to accept input from the user as per the requirement of the program.

Practice Programs:

1. Write a PHP Script to define an array. Find the element from the array that matches the given value using the appropriate search function.
2. Write a PHP script to count the total number of vowels (a,e, i,o,u) from the string. Show the occurrences of each vowel from the string.
3. Write PHP program to perform the following operations on Indexed Array:
   a) Check the array element is positive or negative
   b) Calculate the average of array elements
   c) Calculate the sum of array elements

SET A:

1. Write PHP program to perform the following operations on Indexed Array:
   a) Union of two arrays
   b) Traverse the array elements in random order
2. Write a PHP program to perform the following operations on an associative array:
   a) Display the elements of an array along with the keys.
   b) Display the size of an array
   c) Delete an element from an array from the given index.
   d) Reverse the order of each element's key-value pair

e) Traverse the elements in an array in random order.

3. Write a PHP Script for the following:
   a) Declare and Display a multidimensional Array.
   b) Search and display a specific element from a Multidimensional array.

SET B:

1. Write a PHP script to perform the following operations on string :
   i) Compare string 2 with string3.
   ii) Convert all the strings to Upper case
   iii) Convert all the strings to Lowercase
2. Write a PHP script to perform the following operations on string :
   i) Convert each word of a string to Lowercase and Uppercase.
   ii) Find the first and last occurrence of string2 in string1.
3. Write a menu-driven program in PHP to perform the following operations on associative arrays:
   i) Sort the array by values (changing the keys) in ascending, descending order.
   ii) Also, sort the array by values without changing the keys.
   iii) Find the intersection of two arrays.
   iv) Find the union of two arrays.

SET C:

1. Write a PHP script to perform the following operations on string :
   i) Replace the string2 by string3 in string1.
   ii) Reverse and display the string.

Assignment Evaluation

0: Not Done [ ] **1: Incomplete [ ]** **2: Late Complete [ ]**
3: Need Improvement [ ] **4: Complete [ ]** **5: Well Done [ ]**

Signature of Instructor

Assignment 4: Functions, Class, and Object

Functions

A function is a block of code that performs a specific task. It can be called from anywhere from the program. It takes zero or any number of parameters and does some processing and returns a value.

PHP Built-in Functions

Name	Use	Example
echo	This construct is used to display many values at once on the screen.	echo “Hello”; echo (“Hello”);
print()	It prints data to the screen	print (“Hello”);
print_r()	It prints the contents of arrays and objects.	<?php $array = array(1, 2, 3); print_r($array); ?> **Output:** Array ([0] => 1 [1] => 2 [2] => 3)
var_dump()	It returns the value and data type of a given variable.	<?php $a=50; echo var_dump($a); ?> **Output:**Int(50);

isset()	It returns true value, if a given parameter is initialized with a value otherwise it returns false value.	<?php
		$n = 0;
		if (isset($n))
		{
		echo "Variable 'a' is set.”;
		}
		Output: Variable 'a' is set.

unset()	It unsets a variable.	<?php $a = 10; echo "The value of variable 'a' before unset: " . $a . " "; unset($a); echo "The value of variable 'a' after unset: " . $a; ?> **Output:** The value of variable 'a' before unset:10

		The value of variable 'a' after unset:
define()	It is used to define constant.	<?php define ("PI",3.14); echo PI ?> **Output**:3.14.
date()	The date() function formats a local date and time, and returns the formatted date string. Syntax: date(format,timestamp) List of characters commonly used for date: d - Represents the day of the month m - Represents a month Y - Represents a year l - Represents the day of the week	<?php echo "Today is " . date("Y/m/d") . " "; echo "Today is " . date("Y.m.d") . " "; echo "Today is " . date("l"); ?> **Output:** Today is 2021/01/19 Today is 2021.01.19 Today is Tuesday

Defining a user-defined function

While creating a user-defined function its name should be preceded by with keyword **function** and the function code should be put inside { and } braces.

Syntax:

```
Function function_name([parameters])
{
Statements;
}
```

Example:

```
<?php
        /* Defining a PHP Function */ function
    HelloWorld()
    {
    echo "HelloWorld Good Morning!!";
    }
        /* Calling a PHP Function */
    HelloWorld();
?>
```

Output: HelloWorld Good Morning!!

Passing Parameters to Functions

1. Call By Value

When a PHP function is called by value then actual values of variables are not modified if it is modified into the function.

Example:

```
<? php
        Function addFun($num1, $num2)
        {
                $sum = $num1 + $num2;
        echo "Sum of the two numbers is : $sum";
        }
        addFun(15, 20);
 ?>
```

2. Call By Reference

When a PHP function is called by reference then the actual values of the parameters are modified by the function.

Example:

```
<?php
        functionaddFun(&$num1, &$num2)
        {
                $sum = $num1 + $num2;
        echo "Sum of the two numbers is : $sum";
        }
        addFun(15, 20);
 ?>
```

Default Parameter

If we do not pass any value to the function then the function uses a default value called default parameter.

Example:

```
<?php
        Function setHeight($maxheight=100)
        {
        echo "The height is : $maxheight<br>";
        }
        setHeight(350);
        setHeight(); // will use the default value of 100
?>
```

Output:

```
The height is: 350
The height is:100
```

Classes and Objects:

PHP supports to the object oriented programming concepts.

Class:

It is user defined data type. It is a collection of data members and functions as a single unit.

A class can be defined as:

Example:

```
<?php
    class Car
    {
        /* Member variables */
    var $price;
        /* Member functions */
    function setPrice($par)
    {
          $this->price = $par;
    }
    Function getPrice()
    {
    echo $this->price ."<br/>";
    }
  }
?>
```

Object:

Any real or runtime entity is called an object. Objects are also known as instance.

Creating Objects in PHP

After defining a class, an object of that class can be created. It can be done by using a new operator as follow:

Example:

```
Object_name=new Class_Name;
$Car1=new Car;
```

For accessing data member and member functions of a class, an object is used.

Example:

```
$Car1->setPrice(5);
```

Practice Programs:

1. Write a PHP script to calculate the area and volume of a cylinder using a function.
2. Write a PHP Script to display the sum and average of array elements(Using predefined functions)
3. Write a PHP script to calculate the factorial of a number using a function.

SET A:

1. Write a PHP script to calculate x^y using a function.
2. Write a PHP script to define a function EvenOdd, which will display even and odd numbers between 1 to 50.
3. Write a PHP script to define a function Maximum, which will accept 3 numbers as parameters and returns a maximum of 3 numbers.
4. Write a PHP script to swap two numbers using a function (Use Call by value and Call by reference)

SET B:

1. Write a PHP Script to create a class Fruit that contains data members as Name, Color and Price. Write a member function to accept and display details of Fruit.
2. Write a PHP Script to create a class Student that contains data members as Roll_Number, Stud_Name, and Percentage. Write member functions to accept Student information.
3. Write a PHP Script to create a class Book (Book_id, Book_name, Publication, Author, Book_price). Write a member function to accept and display Book details.

SET C:

1. Write a PHP script to define a function "DisplayDay", which will display the day of the current date.
2. Write a PHP script to perform arithmetic operations on two numbers. Write a PHP function to display the result. (Use the concept of function and default parameters)

Assignment Evaluation

0: Not Done [ ] **1: Incomplete [ ]** **2: Late Complete [ ]**
3: Need Improvement [ ] **4: Complete [ ]** **5: Well Done [ ]**

Signature of Instructor

Assignment 5: Working with forms

Processing a Form's Data:

HTML forms are used to send the user information to the server and returns the result to the browser. For example, if you want to get the details of visitors to your website, and send them good thoughts, you can collect the user information employing form processing. Then, the information can be validated either at the client-side or on the server-side. The final result is sent to the client through the respective web browser. To create a HTML form, the following **form** tag should be used.

- Action
- Method

Form processing contains a set of controls through which the client and server can communicate and share information. The controls used in forms are:

- Text
- Textarea
- Dropdown
- Radio
- Checkbox
- Buttons

PHP methods used in form processing are:

- **$_GET[]:** It is used the retrieve the information from the form control through the parameters sent in the URL. It takes the attribute given in the URL as the parameter.
- **$_POST[]:** It is used the retrieve the information from the form control through the HTTP POST method. It takes the name attribute of the corresponding form control as the parameter.

Example Using POST Method

stud.html

```
<html>
<body>
<form method=POST action="stud.php">
        RollNo :          <input type=text name=rno><br>
        Student Name : <input type=text name=sname><br>
        Percentage :     <input type=text name=per><br>
                        <input type=submit value=submit name=submit>
</form>
</body>
</html>
```

stud.php

```
<?php
```

```
        echo $_POST['rno'];
        echo $_POST['sname'];
        echo $_POST['per'];
?>
```

Example Using GET Method

stud.html

```
<html>
<body>
<form method=GET action="stud.php">
        RollNo :          <input type=text name=rno><br>
        Student Name : <input type=text name=sname><br>
        Percentage :     <input type=text name=per><br>
                         <input type=submit value=submit>
</form>
</body>
</html>
```

stud.php

```
<?php
        echo $_GET['rno'];
        echo $_GET['sname'];
        echo $_GET['per'];
?>
```

PHP function used for form processing:

- **isset():** This function is used to determine whether the variable or a form control is having a value or not.

Example using isset()

stud.html

```
<html>
<body>
<form method=POST action="stud.php">
        RollNo :          <input type=text name=rno><br>
        Student Name : <input type=text name=sname><br>
        Percentage :     <input type=text name=per><br>
                         <input type=submit value=submit>
</form>
</body>
</html>
```

stud.php

```
<?php

if (isset($_GET[‘submit’]))
{
        if((!isset($_GET[‘rno’])) ||(!isset($_GET[‘sname’]))|| (!isset($_GET[‘per’])))
        {
                Echo “Please fill all the required fields”;
        }
}
else
{
        echo $_GET[‘rno’];
        echo $_GET[‘sname’];
        echo $_GET[‘per’];
}
?>
```

Self ProcessingPage :

Selfprocessing page means one PHP can be used to both generate a form and process it. PHP_SELF variable is used for self processing page.

PHP_SELF variable returns the name and path of the currently executing script. This variable can used in action attribute of the form.

Example:

```
<form method="Get" action="<?php $_SERVER['PHP_SELF'];?>">
```

Example Self processing page

stud.php

```
<html>
<body>
<form method=GET action="<?php $_SERVER['PHP_SELF'];?>">
        RollNo :          <input type=text name=rno><br>
        Student Name : <input type=text name=sname><br>
        Percentage :     <input type=text name=per><br>
                        <input type=submit value=submit>
</form>

<?php

if (isset($_GET[‘submit’]))
{
        if((!isset($_GET[‘rno’])) ||(!isset($_GET[‘sname’]))|| (!isset($_GET[‘per’])))
        {
                Echo “Please fill all the required fields”;
        }
}
else
```

```
{
        echo $_GET[‘rno’];
        echo $_GET[‘sname’];
        echo $_GET[‘per’];
}
?>
</body>
</html>
```

Sticky Forms:

Form remembers the values that are entered in the input fields. For example Google search box. In the sticky form, the results of a query are accompanied by a search form whose default values are those of the previous query.
To create the sticky form, we have to follow 2 steps:

- Step 1: Taking the data sent by the form by using the “GET” or “POST” method: $data= $_GET[“data”];
- Step 2: Settings that data as a value for text fields and selected or checked for other form elements.

Example Sticky Forms
stud.php

```
<html>
<body>
<form method=GET action="<?php $_SERVER['PHP_SELF'];?>">
        Your Name : <input type=text name=sname value=”<?phpecho$_POST[‘sname’] ?>”>
<br>
            <input type=submit value=submit>
</form>

<?php
        if(isset($_GET[‘sname’]))
        {
            echo $_GET[‘sname’];
        }
?>
</body>
</html>
```

Dealing with Checkbox

```
<html>
<body>
<form method=GET action="<?php echo $_SERVER['PHP_SELF']; ?>">

Select ur Choice<br>
```

```
<input type=checkbox name=ch[] value="1" <?php if($_GET['ch']=="1") echo 'checked="checked"';
?>>
Reading

<input type=checkbox name=ch[] value="2" <?php if($_GET['ch']=="2") echo 'checked="checked"';
?>>
Dancing<br>

<input type=Submit name="S" value=Click>
</form>

<?php
        if ((isset($_GET['S'])))
        {
                $ch=$_GET['ch'];
                if($ch=="1")
                        echo "Reading";
                else
                        echo "Dancing";
        }
?>

</body>
</html>
```

Dealing with Radio button

```
<html>
<body>
<form method=GET action="<?php echo $_SERVER['PHP_SELF']; ?>">
Select ur Choice<br>
<input type=radio name="r" value="1" <?php if($_GET['r']=="1") echo 'checked="checked"';
?>> Add

<input type=radio name="r" value="2" <?php if($_GET['r']=="2") echo 'checked="checked"';
?>> Sub<br>

<input type=Submit name="S" value=Click>
</form>

<?php
        if ((isset($_GET['S'])))
        {
                $ch=$_GET['r'];
                switch($ch)
                {
                case 1:
```

```
                        $a=$t1+$t2;
                        echo "Addition=$a";
                        break;
                case 2:
                        $a=$t1-$t2;
                        echo "Sub=$a";
                        break;
                }
        }
?>

</body>
</html>
```

Retrieving values from List:

```
<html>
<body>
<form method=GET action="<?php echo $_SERVER['PHP_SELF']; ?>">
Select ur Choice<br>
<select name=m>
        <option value="R" <?php if($_GET['m']=="R") echo 'selected="selected"'; ?>>Reading
        <option value="D" <?php if($_GET['m']=="D") echo 'selected="selected"'; ?>>Dancing
</select>
<input type=submit name=S value=Click>
</form>

<?php
        if ((isset($_GET['S'])))
        {
                $ch=$_GET['m'];
                echo $ch;

        }
?>

</body>
</html>
```

Validating and Restricting data:

Different strategies for validating form data are,

- Fields should not be empty
- Check the length of the data entered by the user.
- Check the type of data entered by the user.
- Check specific conditions for form fields

Functions for Validating and Restricting data are,

- **Empty(varName):** it is used to check whether a variable is empty or not.

- **isset():** This function is used to determine whether the variable or a form control is having a value or not.
- **filter_var()** function filters a variable with the specified filter.

  Syntax:

  filter_var(var, filtername, options)

  Parameters: This function accepts three parameters and is described below:
  1. **var** : It is the required field. It denotes the variable to filter.
  2. **Filter name** : It is used to specify the ID or name of the filter to use. Default is FILTER_DEFAULT, which results in no filtering. It is an optional field.
  3. **options** : It is used to specify one or more flags/options to use. Check each filter for possible options and flags. It is also an optional field.

  Return Value: It returns the filtered data on success, or FALSE on failure.

  Filter names are,
  - FILTER_VALIDATE_INT: to check if the variable is an integer or not
  - FILTER_VALIDATE_IP: to check if the variable is a valid IP address or not.
  - FILTER_VALIDATE_EMAIL: to check if the variable is a valid email address or not.
  - FILTER_VALIDATE_URL: to check if the variable is a valid URL or not.

Example using empty()

```
<html>
<body>
<form method=GET action="<?php echo $_SERVER['PHP_SELF']; ?>">
Search<input type=text name="t1" value="<?php if(isset($_GET['t1'])) echo $_GET['t1']; ?>">
        <input type=Submit name="s" value=Click>
</form>

<?php
        if ((isset($_GET['t1'])))
        {
                $t1=$_GET['t1'];
                if ((!empty($t1)))
                {
                        echo $t1;
                }
                else
                {
                        echo "enter the value in textbox";
                }
        }
?>
</body>
</html>
```

Example using filter_var :

FILTER_VALIDATE_INT

```
<?php
$n = 200;
if (filter_var($n, FILTER_VALIDATE_INT) === 0 ||
   !filter_var($int, FILTER_VALIDATE_INT) === false)
{
   echo("Integer is valid");
}
else
{
   echo("Integer is not valid");
}
?>
```

FILTER_VALIDATE_IP:

```
<?php

$ip = "129.0.0.1";
  if (!filter_var($ip, FILTER_VALIDATE_IP) === false)
{
   echo("$ip is a valid IP address");
}
else
{
   echo("$ip is not a valid IP address");
}
 ?>
```

FILTER_VALIDATE_EMAIL:

```
<?php
$email = "abc@gmail.com";
if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false)
{
   echo("$email is a valid email address");
}
else
{
   echo("$email is not a valid email address");
}
?>
```

FILTER_VALIDATE_URL:

```
<?php
  $url= "https://www.google.com";
if(!filter_var($url, FILTER_VALIDATE_URL) === false)
```

```
{
   echo("$url is a valid URL");
}
else
{
   echo("$url is not a valid URL");
}
?>
```

Practice Programs:

1. To design an application that works as a simple calculator using PHP. (use isset()).
2. Write a PHP script to check PAN number entered by the customer is valid or not and display an appropriate message.
3. Write a PHP script to check mobile number entered by the user is valid or not and display an appropriate message.

SET A:

1. Write a PHP script to accept font name, background color, and welcome message on 1st page. Display the welcome message with the given font and background color on the next page.
2. Write a PHP program to accept name, address, pincode, gender information. If any field is blank display error messages “all fields are required”.
3. Write a PHP script to accept employee details (name, address) and earning details (basic, DA, HRA). Display employee details and earning details in the proper format.

SET B:

1. Write a PHP script to accept customer name and the list of product and quantity on the first page. On the next page display the name of the customer, name of the products, rate of the product, quantity, and total price in table format.
2. Write HTML code to design multiple choice question paper for PHP subject. Display question wise marks and total marks received by the student in table format.
3. Write a PHP script to accept student name and list of programming languages (using drop down box) and display it on the next page in the proper format.
4. Write a PHP script to accept user name, email address and age. If data entered by the user is valid then display it on the next page otherwise display the appropriate message(use filter_var()).

SET C:

1. A web application that takes name and age from an HTML page. If the age is less than 18, it should send a page with "Hello <name>, you are not authorized to visit the site" message, where <name> should be replaced with the entered name. Otherwise, it should send a "Welcome <name> to this site" message.

Assignment Evaluation

0: Not Done [ ]	**1: Incomplete [ ]**	**2: Late Complete [ ]**
3: Need Improvement [ ]	**4: Complete [ ]**	**5: Well Done [ ]**

Signature of Instructor

Assignment 6: Session and Cookies

Cookies:

A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.
A cookie is created with the setcookie() function.

setcookie(name[, value, expire, path, domain, secure, httponly]);

where,

- **name:** A unique name for a particular cookie. You can have multiple cookies with different
- names and attributes.
- **value :** It is used to set the value of the cookie
- **expire** : The expiration date for this cookie. If no expiration date is specified, the browser
- saves the cookie in memory and not on disk. When the browser exits, the cookie disappears.
- The expiration date is specified as the number of seconds.
- **path :**It is used to specify the path on the server for which the cookie will be available.
- **domain** :It is used to specify the domain for which the cookie is available.
- **secure** : It is used to indicate that the cookie should be sent only if a secure HTTPS connection exists.

```
<?php
        //Creating a cookie
        $cookie_name = "user";
        $cookie_value = "abc";
        setcookie($cookie_name, $cookie_value, time() + (1* 24 * 60 * 60));

        //Checking a Cookie is set or not
        if(!isset($_COOKIE[$cookie_name]))
        {
        echo "Cookie named '" . $cookie_name . "' is not set!";
        }
        else
        {
        echo "Cookie '" . $cookie_name . "' is set!<br>";
        }

        //Accessing Cookie value
        echo "Value is: " . $_COOKIE[$cookie_name];

        // set the expiration date to one hour ago
        setcookie("user", "", time() - 3600);
        echo "Cookie 'user' is deleted.";

        //cookie expire after 1 day
        setcookie($cookie_name, $cookie_value, time() + (1* 24 * 60 * 60));
?>
```

Session:

A session is a way to store information (in variables) to be used across multiple pages. The information is not stored on the user's computer.By default, session variables last until the user closes the browser.Session variables hold information about one single user and are available to all pages in one application

- The first step is to start up a session. After a session is started, session variables can be created to store information. The PHP **session_start()** function is used to begin a new session. It also creates a new session ID for the user.
- **The second step is to set Session variables using** PHP global variable: **$_SESSION.**

```
<?php
        // Start the session
        session_start();

        // Set session variables
        $_SESSION["favcolor"] = "green";
        $_SESSION["favanimal"] = "cat";
        echo "Session variables are set.<br>";

        // access session data
        echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
        echo "Favorite animal is " . $_SESSION["favanimal"] . "<br>";
        print_r($_SESSION);

        // to change a session variable, just overwrite it
        $_SESSION["favcolor"] = "yellow";
        print_r($_SESSION);

        // remove all session variables
        session_unset();
        if(($_SESSION["favcolor"]!=0) && ($_SESSION["favanimal"]!=0))
                print_r($_SESSION);
        else
                echo "Session variables are unset.<br>";

        // destroy the session
        session_destroy();
        print_r($_SESSION);
        echo "Session variables are destroyed.<br>";

?>
```

Practice Programs:

1. A web application that lists all cookies stored in the browser on clicking "list cookies" button,add cookies if necessary.

2. Write a PHP program to store the current date-time in a COOKIE and display the 'Last visited on'date-time on the web page upon reopening of the same page.
3. Write a script to keep track of a number of times the web page has been accessed using the session.

SET A:

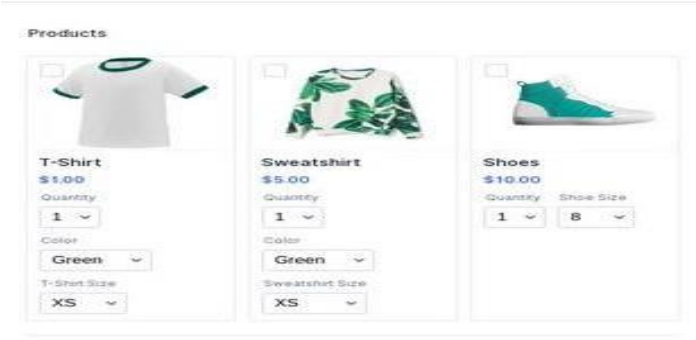
1. Write PHP program to store student information like Seat number, name, and class. On the second page, accept marks of the subject PHP, DS, CPP, and RDBMS. Display Result in table format on the third page (use cookies).
2. Write a PHP script to accept username and password. If in the first three chances, username and password entered is correct, then display the welcome message on the second form, otherwise display an error message.
3. Write a PHP script to accept font style, font size, font color, background color using a cookie. Display selected values on the next second page and actual implementation on the third web page.

SET B:

1. Create an online flight registration form. On the first page accept name, address, birthdate, and mobile number. On the second page accept flight details (flight name, source, destination, departure date-time and charges). If the user doesn't enter information within a specified time limit, expire his session and give a warning otherwise display details using sessions on the third page**.**
2. Create a form to accept patient details like name, address birthdate, and mobile number. Once thePatient information is accepted, and then accepts health details like medicare number, health fund and critical information. Display patient details and health details on the next form.
3. Write a PHP script to create an inventory management system. On the first page accept the highest sold product details like product name, total quantity and total sold. On the second page accept the latest sales details like product name, date and total sale. Display highest sold product details in one table and latest sales details in another table on the third page.

SET C:

1. Write a PHP script to create an online shopping form. On the first page accept customer name, email address, shipping address, mode of payment. Design the Second page as given below. And the third page should display a bill, which consists of customer details and purchase details in the proper format.



Assignment Evaluation

0: Not Done [ ] **1: Incomplete [ ]** **2: Late Complete [ ]**
3: Need Improvement [ ] **4: Complete [ ]** **5: Well Done [ ]**

Signature of Instructor

Assignment 7: Working with the Database

Database :

It is a collection of inter-related data that helps in efficient retrieval, insertion, and deletion of data from the database and organizes the data in the form of tables, views, schemas, reports, etc. The basic functions used in PHP for database connection are,

Function	Description	Example
mysql_connect(server, user, pwd)	It opens a database connection and returns the connection on success, or FALSE and an error on failure.	$con=mysql_connect("localhost","root","");
mysql_select_db(db name)	It is used to change the default database for the connection.	mysql_select_db("sybba");
mysql_query(query);	It executes a query on a MySQL database.This function returns the query handle for SELECT queries, TRUE/FALSE for other queries, or FALSE on failure.	$sql=”Select * from stud” $r=mysql_query($sql);
mysql_fetch_array(result,resulttype);	function fetches a result row as an associative array, a numeric array, or both. result=Required. resulttype=Optional. Specifies what type of array that should be produced. Values are, MYSQL_ASSOC, MYSQL_NUM, MYSQL_BOTH	mysql_fetch_array($r, MYSQL_ASSOC);
mysql_close(connection);	The mysql_close() function closes MySQL connection.This function returns TRUE on success, or FALSE on failure.	mysql_close($con);

1. The basic steps to create a MySQL database using PHP are:
   - Establish a connection to the MySQL server from your PHP script.
   - If the connection is successful, write a SQL query to create a database and store it in a string variable.
   - Execute the query.
   - Close the connection

```
<?php
        $con=mysql_connect("localhost","root","");
```

```
        if(!$con)
        {
        die("unable to connect");
    }
    $sql="create database sybba";
$r=mysql_query($sql);
if(! $r)
    {
        die("could not create database");
    }
echo "Database created successfully";
mysql_close($con);
?>
```

2. The basic steps to create a MySQL table using PHP are:
    - Establish a connection to the MySQL server from your PHP script.
    - If the connection is successful, then select the database.
    - Write a SQL query to create a table and store it in a string variable.
    - Execute the query.
    - Close the connection

```
<?php
        $con=mysql_connect("localhost","root","");
        if(!$con)
        {
        die("unable to connect");
    }
mysql_select_db("sybba");

    $sql="create table stud(rnoint, snamevarchar(20), per int)";
$r=mysql_query($sql);
if(! $r)
    {
        die("could not create table");
    }
    echo "Table created successfully";
mysql_close($con);
```

3. The basic steps to manipulate MySQL table using PHP are:
    - Establish a connection to the MySQL server from your PHP script.
    - If the connection is successful, then select the database.
    - Write aninsert/update/delete query to manipulate and store it in a string variable
    - Execute the query.

- Close the connection

```
<?php
        $con=mysql_connect("localhost","root","");
        if(!$con)
        {
        die("unable to connect");
      }
mysql_select_db("sybba");

      $sql="insert into stud values(1,'Neeta',84)";
$r=mysql_query($sql);
if(! $r)
      {
          die("not inserted");
      }
      echo "record added successfully";
mysql_close($con);
?>
```

4. The basic steps to fetchdata from MySQL table using PHP are:
    - Establish a connection to the MySQL server from your PHP script.
    - If the connection is successful, then select the database.
    - Write a select query and store it in a string variable
    - Execute the query
    - Display data using a while loop.
    - Close the connection

```
<?php
        $con=mysql_connect("localhost","root","");
        if(!$con)
        {
        die("unable to connect");
        }
        mysql_select_db("sybba");
        $result=mysql_query("select * from stud");
while($col=mysql_fetch_array($result,MYSQL_NUM))
        {
                echo "Rollno=".$col[0]."<br>";
                echo "Name=".$col[1]."<br>";
                echo "Per =".$col[2]."<br>";
        }
        mysql_close($con);
?>
```

Practice Programs:

1. Consider the following entities and their relationships
   Company (c_no, c_name, c_city, c_share_value)
   Person (p_no, p_name, p_city, p_ph_no)
   Relationship between Company and Person is many-to-many with descriptive attribute no_of_shares.
   Using the above database, write a PHP script to display person wise share detailsin tabular format.

2. Consider the following entities and their relationships
   Customer (c_no, c_name, c_city, c_ph_no)
   Ticket (t_no, booking_date, fare, traveling_date)

The relationship between Customer and Ticket is one-to-many. Create a RDB in 3 NF for the above.
Using the above database, write a PHP script to accept date and display,
1) The total fare collected from customers on a given date.
2) Ticket details booked by the customer.

SET A:

1. Write a PHP script to create an employee table using attributes employee number, employee name, address joining date and salary. If a table is created then display the appropriate message otherwise end the PHP script.
2. Write a PHP script to accept account details (account number, account type and balance). Store these details in the account table and display an appropriate message.
3. Write a PHP script to accept product number from the user. Update the price of the product and display an appropriate message.

SET B:

1. Consider the following entities and their relationships.
   Employee (eno, ename, sal)
   Project (pno, pname,duration)
   Employee and Project are related with a many-many relationship. Create a RDB in 3 NF for the above.
   Using the above database write a PHP script to accept the project name. Display the name of the employees and the duration of the project.
2. Consider the following entities and their relationships.
   Train(t_no, t_name)
   Passenger (p_no, p_name, addr, age)
   The relationship between Train and Passenger is many-to-many with descriptive attribute date,seat_no and amt. Create a RDB in 3 NF for the above.
   Using the above database write a PHP script to accept a date. Display train details having maximum passenger for a given date.
3. Consider the following entities and their relationships.
   Crop (c_no, c_name, c_season, pesticides)

Farmer (f_no, f_name, f_location)

The relationship between Crop and Farmer is many-to-many with descriptive attribute year. Create a RDB in 3 NF for the above.

Using the above database write a PHP script to accept crop name and year value. Display total number of farmers harvesting given crop in a given year.

SETC:

1. Consider the following entities and their relationships. Client (c_no, c_name, c_addr, birth_date)

   Policy_info (p_no, p_name, maturity_amt, prem_amt, policy_term)

   The relationship between Client and Policy_info is many-to-many with descriptive attribute date_of_purchase.Create a RDB in 3NF for the above.

   Using the above database write a PHP script to display policy details of a given client for a given year in the following format.

Client Name : **Year:**

Policy Name	Maturity Amount	Premium Amount	Policy Term	Date of Purchase

Assignment Evaluation

0: Not Done [ ] **1: Incomplete [ ]** **2: Late Complete [ ]**
3: Need Improvement [ ] **4: Complete [ ]** **5: Well Done [ ]**

Signature of Instructor