B.L.D.E. Association's

S.B.ARTS & K.C.P SCIENCE COLLEGE
VIJAYAPUR

M.Sc. Computer Science

A REPORT ON

Bridge Course

Entitled

"Technical, Theoretic and Practical approaches in Higher Education"

For
M.Sc (CS)-I Sem Students

2019-2020

Resource Person

Prof(Smt) S.D.Patil
Prof(Smt) R.D.Joshi
Prof S.V. Vambase
Prof Pavankumar Mahindrakar

S.B.ARTS & K.C.P SCIENCE COLLEGE
VIJAYAPUR

M.Sc(CS) Programme

NOTICE

It is here by informed to all the M.Sc(CS) I Semester students that there will be Bridge Course from 13th Aug 2019 - 17th Aug 2019 .So all of you should attend and get the benefit.

Co-ordinator
M.Sc. (C.S.) Programme
S.B.Arts & K.C.P.Science College,
Vijayapur.

IQAC, Co-ordinator
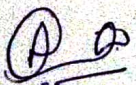S.B.Arts & K.C.P.Science College,
Vijayapur.

Principal
S.B.Arts & K.C.P.Science College
Vijayapur.

B.L.D.E.Association's

S.B.Arts and K.C.P Science College, Bijapur

M.Sc(CS) Programme

M.Sc(CS)– I Semester 2019-2020

Bridge Course Time Table

Date	Time	Subject
13/08/2019	11.00 am to 2.00 pm	Introduction to Programming & its Techniques.
14/08/2019	11.00 am to 2.00 pm	Introduction of Operating System & its Services.
16/08/2019	11.00 am to 2.00 pm	Mapping of Hardware with Logic Gates.
17/08/2019	11.00 am to 2.00 pm	Data Structure

Co-ordinator
M.Sc. (C.S.) Programme
S.B.Arts & K.C.P.Science College,
Vijayapur.

IQAC, Co-ordinator
S.B.Arts & K.C.P.Science College,
Vijayapur.

Principal
S.B.Arts & K.C.P.Science College,
Vijayapur.

About Bridge Course

The Bridge Course is aimed to act as a buffer for the new entrants, with an objective to provide adequate time for the transition to hard-core engineering courses. During this interaction of bridge course week with the faculty and their classmates, the students will be equipped with the knowledge and the confidence needed to take on bigger challenges as future engineers of this country.

Objectives: To act as a buffer for the new entrants.

- To provide adequate time for the transition to hard-core engineering courses.

- Focus on fostering a strong sense of ethical judgment and moral fortitude.

- Applications based self-learning and intermingling of a large cross section of students from vastly

  varying backgrounds.

- A breather, to prepare themselves before courses for first year engineering commence.

- The students will be equipped with the knowledge and the confidence needed to take on bigger challenges.

- Nurture a deeper understanding of the local and global world and our place in at as concerned citizens of the world.

- Interactive and Active Learning by Doing have been weaved into the Bridge Course.

- Active learning with the help of other students.

M.Sc(CS) Programme

Syllabus
Subject: Introduction of programming and its Techniques

Unit -I : Introduction to Programming Language, History, methodologies and approaches.
Unit-II: Programming technique Algorithm, introduction, abbreviation, procedure for how to write simple tasks, tasks on branching, control and arrays .
Unit-III: Programming technique Flowchart, introduction, ANSI standard symbols of flow chart simple tasks, tasks on branching, control and arrays .

Content

Unit-I : What is data ? definition and example, information definition and example , instructions definition and example, information definition and example , language definition , types of language like HLL, LLL, ALL and MLL and examples. History of language starting with B , c and c++ etc, Methodologies like procedural, mathematics, structure , object oriented , aspect oriented etc then various approach of Top-down and Bottom-Up approach and examples

Unit-II: Definition of algorithm, simple steps how to write an algorithm , write skills of algorithm solved simple algorithms like finding area of circle, triangle, rectangle, simple interest, calculation of total and average of three subjects, algorithms on branching statements like conditional and unconditional statements finding whether given number is positive, or negative etc. algorithms on control statements like counter its definition, increment or decrement operators, and checking the conditions
Like if with goto without goto etc

Unit-III: Definition of Flow chart, various symbols of flowchart like start, stop, input/output, calculation , decision , flow of process, continuation of flow char from one page to another page, terminal, rectangle, parallelogram, diamond (Rhombus) , left, right, up and down arrows, circle with symbols, solved flow charts on simple, conditional and control statements assignments: calculation of circumference of circle, area of rectangle, checking whether given integer is even or odd, smallest, biggest amongst any three numbers, looping tasks on generating natural nos, even nos, fibonacci series ,odd nos, sum of even, odd and natural numbers, etc

S.B.ARTS & K.C.P SCIENCE COLLEGE
VIJAYAPUR

M.Sc(CS) Programme

Syllabus
Subject: Introduction of Operating System and its services

UNIT 1: Overview: Definition of Operating System, Functions of an Operating System.
UNIT 2: Types of Operating System: Batch operating system Time-sharing operating systems, Distributed operating System , Network operating System and Real Time operating System
UNIT 3: Services: Services , Batch Processing. Processes : Process, Program, Process Life Cycle.
UNIT 4: Linux : Components of Linux System, Basic Features , Architecture and Basic Commands.

Content

UNIT 1: Overview

An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.Some popular Operating Systems include Linux Operating System, Windows Operating System, VMS, OS/400, AIX, z/OS, etc.

Definition

An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.

Following are some of important functions of an operating System.

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

UNIT 2:Types of Operating System

Operating systems are there from the very first computer generation and they keep evolving with time. In this chapter, we will discuss some of the important types of operating systems which are most commonly used.

Batch operating system,Time-sharing operating systems,Distributed operating System,Network operating System and Real Time operating System.

UNIT3:Operating System - Services

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system –

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

Batch processing

Batch processing is a technique in which an Operating System collects the programs and data together in a batch before processing starts. An operating system does the following activities related to batch processing –

- The OS defines a job which has predefined sequence of commands, programs and data as a single unit.

- The OS keeps a number a jobs in memory and executes them without any manual information.

- Jobs are processed in the order of submission, i.e., first come first served fashion.

- When a job completes its execution, its memory is released and the output for the job gets copied into an output spool for later printing or processing.

UNIT 4:Operating System - Processes

A process is basically a program in execution. The execution of a process must progress in a sequential fashion.A process is defined as an entity which represents the basic unit of work to be implemented in the system.To put it in simple terms, we write our computer programs in a text file and when we execute this program, it becomes a process which performs all the tasks mentioned in the program.

When a program is loaded into the memory and it becomes a process, it can be divided into four sections — stack, heap, text and data. The following image shows a simplified layout of a process inside main memory –

Program: A program is a piece of code which may be a single line or millions of lines. A computer program is usually written by a computer programmer in a programming language. For example, here is a simple program written in C programming language −

```
#include <stdio.h>

int main() {
  printf("Hello, World! \n");
  return 0;
}
```

A computer program is a collection of instructions that performs a specific task when executed by a computer. When we compare a program with a process, we can conclude that a process is a dynamic instance of a computer program. A part of a computer program that performs a well-defined task is known as an **algorithm**. A collection of computer programs, libraries and related data are referred to as a **software**.

Process Life Cycle

When a process executes, it passes through different states. These stages may differ in different operating systems, and the names of these states are also not standardized.

In general, a process can have one of the following five states at a time.

S.N.	State & Description
1	**Start:** This is the initial state when a process is first started/created.
2	**Ready:** The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run. Process may come into this state after **Start** state or while running it by but interrupted by the scheduler to assign CPU to some other process.
3	**Running:** Once the process has been assigned to a processor by the OS scheduler, the process state is set to running and the processor executes its instructions.
4	**Waiting:** Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available.
5	**Terminated or Exit:** Once the process finishes its execution, or it is terminated by the operating system, it is moved to the terminated state where it waits to be removed from main memory.

Linux

Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Components of Linux System

Linux Operating System has primarily three components

- **Kernel** – Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.

- **System Library** – System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not requires kernel module's code access rights.

- **System Utility** – System Utility programs are responsible to do specialized, individual level tasks.

Basic Features

Following are some of the important features of Linux Operating System.

- **Portable** – Portability means software can works on different types of hardware in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.

- **Open Source** – Linux source code is freely available and it is community based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.

- **Multi-User** – Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.

- **Multiprogramming** – Linux is a multiprogramming system means multiple applications can run at same time.

- **Hierarchical File System** – Linux provides a standard file structure in which system files/ user files are arranged.

- **Shell** – Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs. etc.

- **Security** – Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

Architecture

The architecture of a Linux System consists of the following layers –

- **Hardware layer** – Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).

- **Kernel** – It is the core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.

- **Shell** – An interface to kernel, hiding complexity of kernel's functions from users. The shell takes commands from the user and executes kernel's functions.

- **Utilities** – Utility programs that provide the user most of the functionalities of an operating systems.

M.Sc(CS) Programme

Syllabus
Mapping of Computer Hardware with logic gates

UNIT 1: Processor Architecture: Introduction to Processor, architecture and applications.
UNIT 2: Components: components-ALU, Coprocessor, clock signal, synchronzation of modules.
UNIT 3: Functionality: Multiplexer, Demultiplexer, Address bus and Data bus, DMA.
UNIT 4: Mapping: significance of logic gate in design of memory and processor.

Content

UNIT 1:
　　History, processor: 4-bit, 8-bit, 16-bit 32 bit and 64-bit, Heat sink, Intel and AMD processor comparison.

UNIT 2:
　　Arithmetic and Logic operations, Math coprocessor, DMA, clock signal,

UNIT 3:
　　Significance of Multiplexer, Demultiplexer and its applications. Address bus, data bus and multiplexed Address/Data bus. Direct Memory access.

UNIT 4:
　　The logic gates as building block for - processor, memory, clock, multiplexer, demultiplexer, encoder, decoder, CPU register

S.B.ARTS & K.C.P SCIENCE COLLEGE
VIJAYAPUR

M.Sc(CS) Programme

Syllabus
Data Structure

Unit 1: Data, Definition, Data types, Built-in Data Type, Derived Data Type, Basic types of Data Structures, Algorithm, Complexity

Unit 2: Basic Operations, Classification Data Structures, Arrays Linked List

Unit 3: Stack

Unit 4: Queue, Trees, Heap, Dictionary, Graph

Unit 5: Real life applications of data structures

Content

Data Definition defines a particular data with the following characteristics.

- Atomic: Definition should define single concept
- Traceable: Definition should be able to mapped to some data element.
- Accurate: Definition should be unambiguous.
- Clear and Concise: Definition should be understandable.

Data Object

Data Object represents an object having a data.

Data Type

Data type is a way to classify various types of data such as integer, string, etc. which determines the values that can be used with the corresponding type of data, the type of operations that can be performed on the corresponding type of data. There are two Data type_

- Built-in Data Type
- Derived Data Type

Built-in Data Type

Those data types for which a language has built-in support are known as Built-in Data types. For example, most of the languages provide the following built-in data types.

- Integers
- Boolean (true, false)
- Floating (Decimal numbers)
- Character and Strings

Derived Data Type

Those data types which are implementation independent as they can be implemented in one or the other way are known as derived data types. These data types are normally built by the combination of primary or built-in data type and associated operations on them. For example

- List

Depth-first search

Converting a decimal number into a binary number

Queue

Queue is a first-in-first-out data structure. The element that is added to the queue data structure first, will be removed from the queue first. Desuetude, priority queue, and circular queue are the variants of queue data structure. Queue has the following application uses:

- Access to shared resources (e.g., printer)
- Multiprogramming
- Message queue

Trees

Tree is a hierarchical data structure. The very top element of a tree is called the root of the tree. Except the root element every element in a tree has a parent element, and zero or more children elements. All elements in the left sub-tree come before the root in sorting order, and all those in the right sub-tree come after the root.

Heap

Heap is a binary tree that stores a collection of keys by satisfying heap property. Max heap and min heap are two flavours of heap data structure. The heap property for max heap is: each node should be greater than or equal to each of its children. While, for min heap it is: each node should be smaller than or equal to each of its children. Heap data structure is usually used to implement priority queues.

Dictionary

Dictionary is a data structure that maintains a set of items indexed on basis of keys. Dictionary stores data in form of key-element pairs.

Hash Table

Hash Table is again a data structure that stores data in form of key-element pairs. A key is a non-null value which is mapped to an element. And, the element is accessed on the basis of the key associated with it. Hash table is a useful data structure for implementing dictionary.

Graph

Graph is a networked data structure that connects a collection of nodes called vertices, by connections, called edges. An edge can be seen as a path or communication link between two nodes. These edges can be either directed or undirected. If a path is directed then you can move in one direction only, while in an undirected path the movement is possible in both directions.

Introduction to Data Structures

Data Structure is a way of collecting and organising data in such a way that we can perform operations on these data in an effective way. Data Structures is about rendering data elements in terms of some relationship, for better organization and storage. For example, we have data player's name "Virat" and age 26. Here "Virat" is of **String** data type and 26 is of **integer** data type.

Basic types of Data Structures

As we discussed above, anything that can store data can be called as a data strucure, hence Integer, Float, Boolean, Char etc, all are data structures. They are known as Primitive Data Structures.

- Linked List
- Tree
- Graph
- Stack, Queue etc.

What is Algorithm

An algorithm is a finite set of instructions or logic, written in order, to accomplish a certain predefined task. Algorithm is not the complete code or program, it is just the core logic(solution) of a problem, which can be expressed either as an informal high level description as pseudocode or using a flowchart. An algorithm is said to be efficient and fast, if it takes less time to execute and consumes less memory space. The performance of an algorithm is measured on the basis of following properties:

1. Time Complexity
2. Space Complexity

Space Complexity

It's the amount of memory space required by the algorithm, during the course of its execution. Space complexity must be taken seriously for multi-user systems and in situations where limited memory is available.

An algorithm generally requires space for following components:

- **Instruction Space:** It's the space required to store the executable version of the program. This space is fixed, but varies depending upon the number of lines of code in the program.
- **Data Space:** It's the space required to store all the constants and variables value.
- **Environment Space:** It's the space required to store the environment information needed to resume the suspended function.

Time Complexity

Time Complexity is a way to represent the amount of time needed by the program to run to completion.

Time Complexity of Algorithms

Time complexity of an algorithm signifies the total time required by the program to run to completion.

The time complexity of algorithms is most commonly expressed using the big O notation.

Types of Notations for Time Complexity

Now we will discuss and understand the various notations used for Time Complexity.

1. **Big Oh** denotes "*fewer than or the same as*" <expression> iterations.
2. **Big Omega** denotes "*more than or the same as*" <expression> iterations.

3. **Big Theta** denotes "*the same as*" <expression> iterations.
4. **Little Oh** denotes "*fewer than*" <expression> iterations.
5. **Little Omega** denotes "*more than*" <expression> iterations.

Real life applications of data structures:

1. To store a set of programs which are to be given access to a hard disk according to their priority.
2. For representing a city region telephone network.
3. To store a set of fixed key words which are referenced very frequently
4. To represent an image in the form of a bitmap.
5. To implement back functionality in the internet browser.
6. To store dynamically growing data which is accessed very frequently, based upon a key value.
7. To implement printer spooler so that jobs can be printed in the order of their arrival.
8. To record the sequence of all the pages browsed in one session.
9. To implement the undo function in a text editor.
10. To store information about the directories and files in a system.
11. Hash Table - used for fast data lookup - symbol table for compilers, database indexing, caches,Unique data representation\
12. Tree - dictionary, such as one found on a mobile telephone for autocompletion and spell-checking, Parsers, Filesystem
13. Suffix tree - fast full text searches used in most word processors.
14. Stack - undo\redo operation in word processors, Expression evaluation and syntax parsing, many virtual machines like JVM are stack oriented.
15. Queues - Transport and operations research where various entities are stored and held to be processed later ie the queue performs the function of a buffer.
16. Priority queues - process scheduling in the kernel
17. Radix tree - IP routing table
18. BSP tree - 3D computer graphics
19. Graphs - Connections/relations in social networking sites, Routing, networks of communication, data organization etc.
20. Heap - Dynamic memory allocation in lisp

Co-ordinator
M.Sc. (C.S.) Programme
S.B.Arts & K.C.P.Science College,
Vijayapur.

IQAC, Co-ordinator
S.B.Arts & K.C.P.Science College,
Vijayapur.

Principal,
S.B. Arts and KCP Science College
VIJAYAPUR

S.B.Arts & K.C.P. Science College, Vijayapur
M.Sc(CS) - I Semester
Bridge Course

Sl No	Name of the Students	13/08/19	14/08/19	16/08/19	17/08/19
1	HARSHA V HIREMATH				
2	SUHAS S HIRASKAR				
3	GURUDEVI V HIREMATH				
4	SHEETAL R HONAWAD				
5	AKSHAY S PUJAR				
6	NAGMAKOUSAR I MAKANDAR				
7	SRUSHTI S TELI				
8	AKSHATA B PUTHANE				
9	MAHESH M BURKE				
10	NIKITA A HERALGI				
11	TAZEEN PATEL				
12	OMKAR K DENGI				

Co-ordinator
M.Sc. (C.S.) Programme
S.B.Arts & K.C.P.Science College,
Vijayapur.

Principal
S.B.Arts & K.C.P.Science College,
Vijayapur.

IQAC, Co-ordinator
S.B.Arts & K.C.P.Science College,
Vijayapur.