CELEBRATING 25 YEARS
CMRIT
CMR INSTITUTE OF TECHNOLOGY, BENGALURU.
ACCREDITED WITH A+ GRADE BY NAAC

Internal Assessment Test I – December 2022

SOLUTION AND SCHEME OF EVALUATION

Sub:	Computer Organization and Architecture			Sub Code:	21CS34	Branch:	CSE
Date:	02/12/22	Duration:	90 minutes	Max Marks:	50	Sem / Sec:	III / A, B, C

Answer any FIVE FULL Questions — OBE: MARKS | CO | RBT

1 (a) What is performance measurement?
Explain the overall SPEC rating for the computer in a program suite.

[5] | 1 | L1

Performance Measurement – 1 Marks

SPEC Rating – 4 Marks

- T is difficult to compute.
- Measure computer performance using benchmark programs.
- System Performance Evaluation Corporation (SPEC) selects and publishes representative application programs for different application domains, together with test results for many commercially available computers.
- Compile and run (no simulation)
- Reference computer
-

$$SPEC\ rating = \frac{\text{Running time on the reference computer}}{\text{Running time on the computer under test}}$$

$$SPEC\ rating = (\prod_{i=1}^{n} SPEC_i)^{\frac{1}{n}}$$

(b) Consider a computer that has byte addressable memory organized in 32 bit words. A program reads ASCII characters and stores them in successive byte locations, starting at location 1000. Show the contents of the two memory words at location 1000 and 1004 after the name "Johnson" has been read according to
(i) Big Endian
(ii) Little Endian
Note: ASCII code for J – 4 AH, o – 6 FH, h – 68 H, n – 6 EH, s – 73 H

[5] | 1 | L3

Big Endian Approach (2.5 Marks)			
1000	**1001**	**1002**	**1003**
J 4 AH	O 6 FH	H 68 H	N 6 EH
1004	**1005**	**1006**	**1007**
S 73 H	O 6 FH	N 6 EH	

Little Endian Approach (2.5 Marks)			
1000	**1001**	**1002**	**1003**
N 6 EH	H 68 H	O 6 FH	J 4 AH
1004	**1005**	**1006**	**1007**
	N 6 EH	O 6 FH	S 73 H

2

Show how the below expression will be executed in Three address, Two address, One address, and Zero address Processors.

X = (A – B) + (C x D)

- **Three-Address (2.5 Marks)**

```
1.  SUB    R1, A, B        ; R1 ← M[A] - M[B]
2.  MUL    R2, C, D        ; R2 ← M[C] * M[D]
3.  ADD    X, R1, R2       ; M[X] ← R1 + R2
```

- **Two-Address (2.5 Marks)**

```
1.  MOV    R1, A           ; R1 ← M[A]
2.  SUB    R1, B           ; R1 ← R1 - M[B]
3.  MOV    R2, C           ; R2 ← M[C]
4.  MUL    R2, D           ; R2 ← R2 * M[D]
5.  ADD    R1, R2          ; R1 ← R1 + R2
6.  MOV    X, R1           ; M[X] ← R1
```

- **One-Address (2.5 Marks)**

```
1.  LOAD A       ;         AC ← M[A]
2.  SUB    B     ;         AC ← AC - M[B]
3.  STORE T      ;         M[T] ← AC
```

[10]

1

L3

4. LOAD C ; AC ← M[C]
5. MUL D ; AC ← AC * M[D]
6. ADD T ; AC ← AC + M[T]
7. STORE X ; M[X] ← AC

- **Zero-Address (2.5 Marks)**
  1. PUSH A ; TOS ← A
  2. PUSH B ; TOS ← B
  3. SUB ; TOS ← (A - B)
  4. PUSH C ; TOS ← C
  5. PUSH D ; TOS ← D
  6. MUL ; TOS ← (C * D)
  7. ADD ; TOS ← (C*D) + (A-B)
  8. POP X ; M[X] ← TOS

3

How the input output operations are to be performed by the processor? Write a program that reads 2 lines of characters and displays it.

[10] 1 L2

Input / Output Operations – 5 Marks

Program – 5 Marks

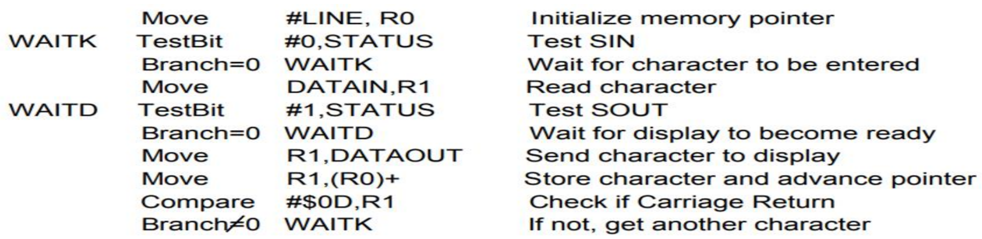
Input:

- When a key is struck on the keyboard, an 8-bit character code is stored in the buffer register DATAIN.
- A status control flag SIN is set to 1 to indicate that a valid character is in DATAIN.
- A program monitors SIN, and when SIN is set to 1, it reads the contents of DATAIN.
- When the character is transferred to the processor, SIN is automatically cleared. Initial state of SIN is 0.

Output:

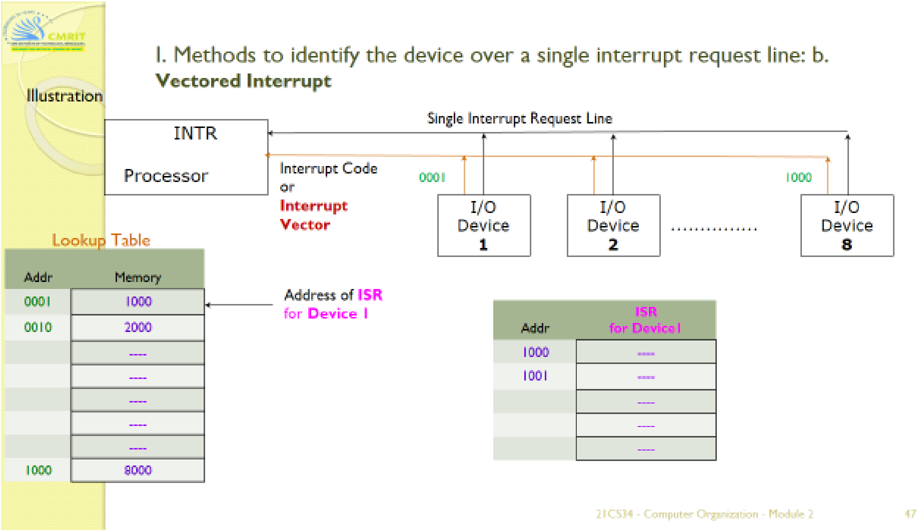
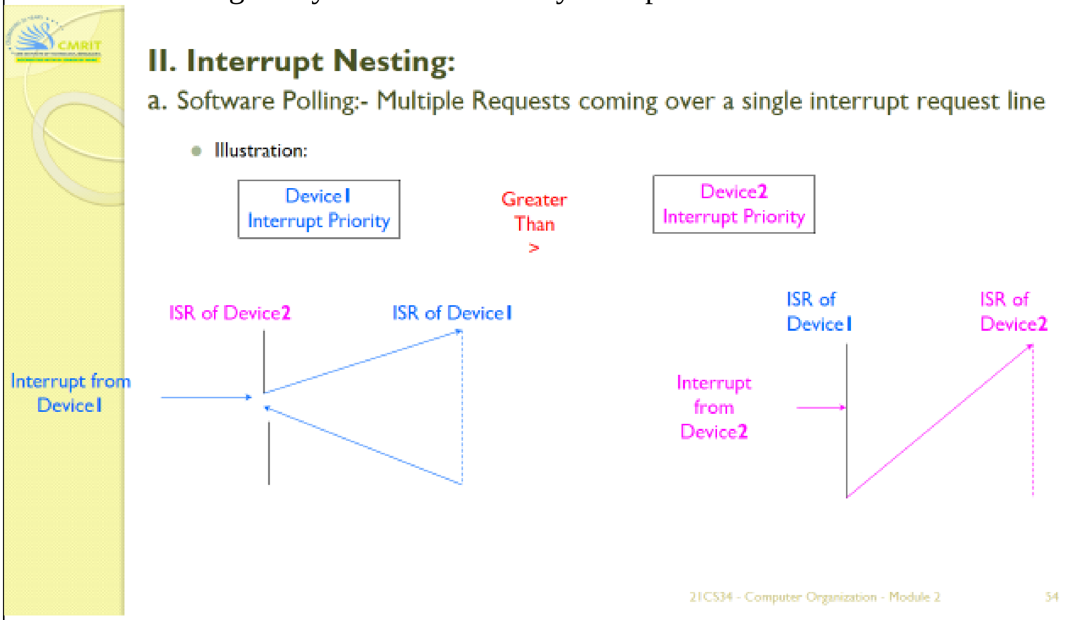
- When SOUT is equal to 1, the display is ready to receive a character.
- A program monitors SOUT, and when SOUT is set to 1, the processor transfers a character code to the buffer DATAOUT.
- Transfer of a character code to DATAOUT clears SOUT to 0.Initial state of SOUT is 1.

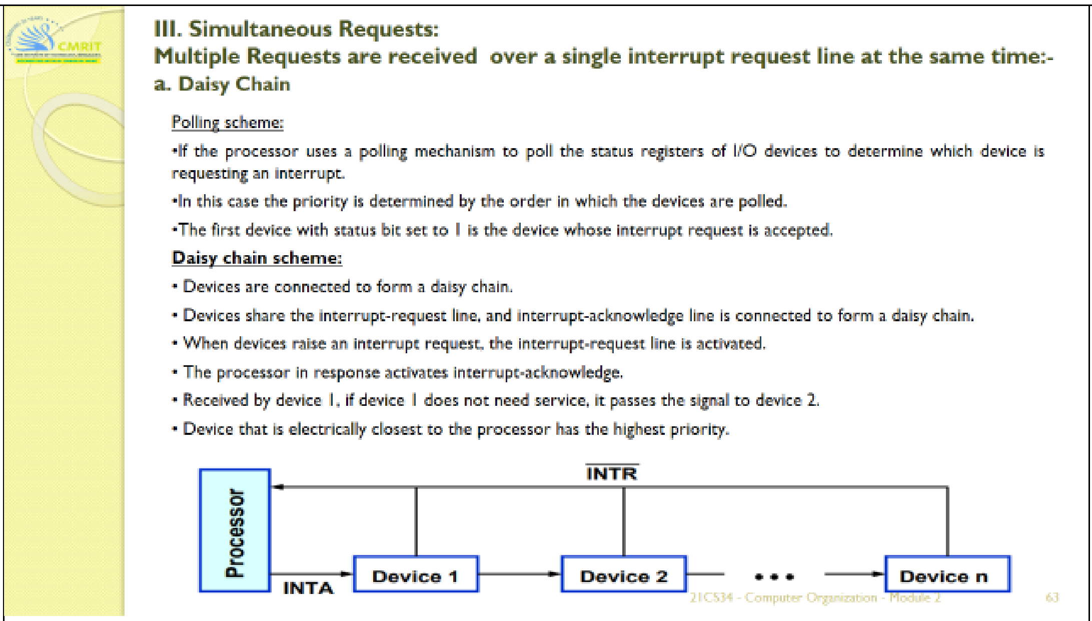
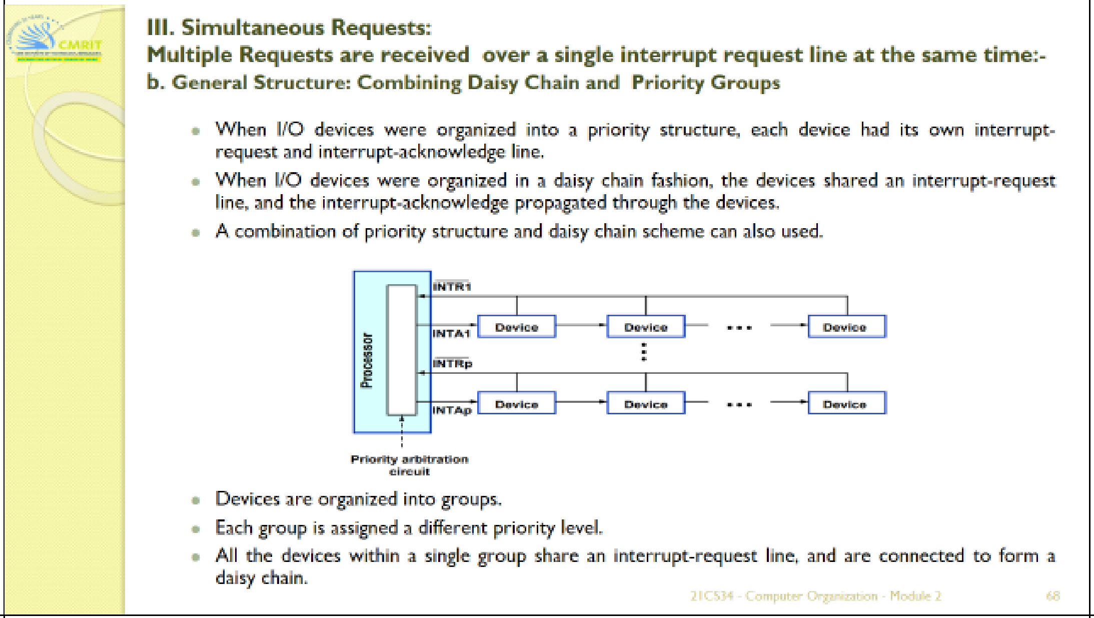
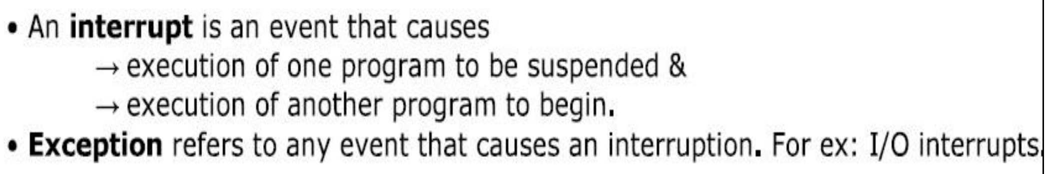
Program:

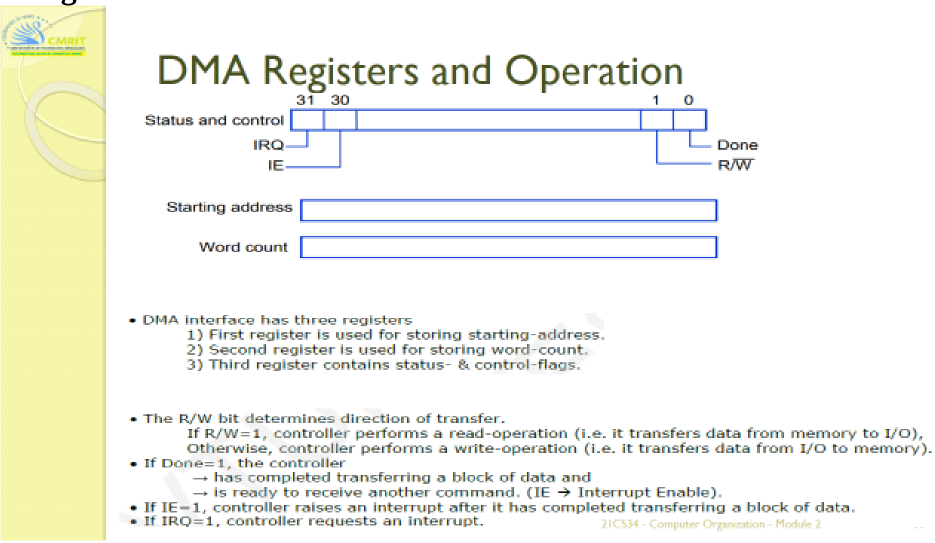


```
         Move       #LINE, R0        Initialize memory pointer
WAITK    TestBit    #0,STATUS        Test SIN
         Branch=0   WAITK            Wait for character to be entered
         Move       DATAIN,R1        Read character
WAITD    TestBit    #1,STATUS        Test SOUT
         Branch=0   WAITD            Wait for display to become ready
         Move       R1,DATAOUT       Send character to display
         Move       R1,(R0)+         Store character and advance pointer
         Compare    #$0D,R1          Check if Carriage Return
         Branch≠0   WAITK            If not, get another character
```

4	What is the effective address of memory operand in each of the following instructions, when the registers R1 and R2 contains the decimal values 1200 and 4600. (i) Load 20(R1), R5 (ii) Move #3000, R5 (iii) Store R5, 30(R1, R2) (iv) Add -(R2), R5 (v) Subtract (R1)+, R5 **5 Questions – Each 2 Marks** (i) Load 20(R1),R5 : This means load 20+R1 into R5 . R1= 1200 , R1 + 20 = 1220 , so R5 have 1220 , Effective address of R5 is 1220. (ii) Move #3000,R5 : This means move value 3000 into R5 , so effective address is part of the instruction whose value is 3000. (iii) Now R5 = 3000 Store R5,30(R1,R2) : This means 30+R1+R2 and store the result into R5. So R5 = 30+1200+4600 = 5830 , so now R5 value is 5830 , the effective address is 5830. (iv) Add -(R2),R5 : This means -1 from R2 value and store the result into R5 . So R5= 4600 - 1 = 4599 , effective address of R5 is 4599 . It is pre decrement addressing. (v) Subtract (R1)+, R5 : This means effective address is contents of R5 so EA = 1200. It is post increment addressing. Effective addresses are 1. 1220 2. 3000 [it is not the effective address , it is the address of the instruction part where 3000 is stored] 3. 5830 4. 4599 5. 1200	[10]	1	L3

5 a	Given that different devices are likely to require different interrupt-service routines, how can the processor identify the device and obtain the starting address of the appropriate routine in each case? Explain **Explanation of Interrupt Code / Interrupt Vector with figure – 2.5 Marks** **Explanation of Lookup Table – 2.5 Marks**  I. Methods to identify the device over a single interrupt request line: b. **Vectored Interrupt** Illustration INTR Processor Single Interrupt Request Line Interrupt Code or Interrupt Vector 0001 1000 I/O Device 1 I/O Device 2 .... I/O Device 8 Lookup Table Addr Memory 0001 1000 0010 2000 ---- ---- ---- ---- ---- 1000 8000 Address of ISR for Device 1 Addr ISR for Device1 1000 ---- 1001 ---- ---- ---- ---- 21CS34 - Computer Organization - Module 2 47	[5]	2	L2
5 b	For a Processor, Multiple Requests are received over a single interrupt request line at the same time. Help the processor to handle this situation with 2 different mechanisms. **Any 2 of the below 3 mechanism with each 2.5 Marks** 1. Software Polling 2. Daisy Chain 3. Combining Daisy Chain and Priority Groups  **II. Interrupt Nesting:** a. Software Polling:- Multiple Requests coming over a single interrupt request line - Illustration: Device1 Interrupt Priority Greater Than > Device2 Interrupt Priority ISR of Device2 ISR of Device1 Interrupt from Device1 ISR of Device1 ISR of Device2 Interrupt from Device2 21CS34 - Computer Organization - Module 2 54	[5]	2	L2

	 **III. Simultaneous Requests:** **Multiple Requests are received over a single interrupt request line at the same time:-** **a. Daisy Chain** Polling scheme: •If the processor uses a polling mechanism to poll the status registers of I/O devices to determine which device is requesting an interrupt. •In this case the priority is determined by the order in which the devices are polled. •The first device with status bit set to 1 is the device whose interrupt request is accepted. **Daisy chain scheme:** • Devices are connected to form a daisy chain. • Devices share the interrupt-request line, and interrupt-acknowledge line is connected to form a daisy chain. • When devices raise an interrupt request, the interrupt-request line is activated. • The processor in response activates interrupt-acknowledge. • Received by device 1, if device 1 does not need service, it passes the signal to device 2. • Device that is electrically closest to the processor has the highest priority. INTR Processor INTA Device 1 Device 2 • • • Device n 21CS34 - Computer Organization - Module 2 63  **III. Simultaneous Requests:** **Multiple Requests are received over a single interrupt request line at the same time:-** **b. General Structure: Combining Daisy Chain and Priority Groups** • When I/O devices were organized into a priority structure, each device had its own interrupt-request and interrupt-acknowledge line. • When I/O devices were organized in a daisy chain fashion, the devices shared an interrupt-request line, and the interrupt-acknowledge propagated through the devices. • A combination of priority structure and daisy chain scheme can also used. Processor INTR1 INTA1 INTRp INTAp Device Device ... Device Device Device ... Device Priority arbitration circuit • Devices are organized into groups. • Each group is assigned a different priority level. • All the devices within a single group share an interrupt-request line, and are connected to form a daisy chain. 21CS34 - Computer Organization - Module 2 68			
6 a	Define Exception. Explain any two kinds of exception **Definition – 1 Mark**  • An **interrupt** is an event that causes → execution of one program to be suspended & → execution of another program to begin. • **Exception** refers to any event that causes an interruption**.** For ex: I/O interrupts. **Any 2 kinds out of the below 3 – each 2 Marks** 1. Recovery from Errors 2. Debugging 3. Privileged Exceptions	[5]	2	L1

	1. Recovery from Errors • These are techniques to ensure that all hardware components are operating properly. • For ex: Many computers include an ECC in memory which allows detection of errors in stored-data. (ECC → Error Checking Code, ESR → Exception Service Routine). • If an error occurs, control-hardware → detects the errors & → informs processor by raising an interrupt. • When exception processing is initiated (as a result of errors), processor. → suspends program being executed & → starts an ESR. This routine takes appropriate action to recover from the error. **2. Debugging** • Debugger → is used to find errors in a program and → uses exceptions to provide 2 important facilities: i) Trace & ii) Breakpoints **i) Trace** • When a processor is operating in trace-mode, an exception occurs after execution of every instruction (using debugging-program as ESR). • Debugging-program enables user to examine contents of registers, memory-locations and so on. • On return from debugging-program, next instruction in program being debugged is executed, then debugging-program is activated again. • The trace exception is disabled during the execution of the debugging-program. **ii) Breakpoints** • Here, the program being debugged is interrupted only at specific points selected by user. • An instruction called Trap (or Software interrupt) is usually provided for this purpose. • When program is executed & reaches breakpoint, the user can examine memory & register contents. **3. Privilege Exception** • To protect OS from being corrupted by user-programs, **Privileged Instructions** are executed only while processor is in supervisor-mode. • For e.g. When processor runs in user-mode, it will not execute instruction that change priority of processor. • An attempt to execute privileged-instruction will produce a **Privilege Exception.** • As a result, processor switches to supervisor-mode & begins to execute an appropriate routine in OS.			
6 b	Explain the concept of DMA along with the registers used in the DMA Interface. **Concept of DMA – 1 Mark** • Direct memory access (DMA) is a method that allows an input/output (I/O) device to send or receive data directly to or from the main memory, without the continuous intervention by the processor. • To transfer large blocks of data at high speed, DMA approach will be used. **DMA Registers – 4 Marks**  DMA Registers and Operation Status and control 31 30 1 0 IRQ IE Done R/$\overline{W}$ Starting address Word count • DMA interface has three registers 1) First register is used for storing starting-address. 2) Second register is used for storing word-count. 3) Third register contains status- & control-flags. • The R/W bit determines direction of transfer. If R/W=1, controller performs a read-operation (i.e. it transfers data from memory to I/O), Otherwise, controller performs a write-operation (i.e. it transfers data from I/O to memory). • If Done=1, the controller → has completed transferring a block of data and → is ready to receive another command. (IE → Interrupt Enable). • If IE=1, controller raises an interrupt after it has completed transferring a block of data. • If IRQ=1, controller requests an interrupt. 21CS34 - Computer Organization - Module 2	[5]	2	L1

CI CCI HoD

CO PO Mapping

Course Outcomes		Modules covered	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	Explain the organization and architecture of computer systems with machine instructions and programs	**1**	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CO2	Analyze the input/output devices communicating with computer system	**2**	3	-	-	-	-	-	-	-	-	-	-	-	2	-	-	-
CO3	Demonstrate the functions of different types of memory devices	**3**	2	2	2	-	-	-	-	-	-	-	-	-	-	2	-	-
CO4	Apply different data types on simple arithmetic and logical unit	**4**	3	-	-	-	-	-	-	-	-	-	-	-	-	2	-	-
CO5	Analyze the functions of basic processing unit, Parallel processing and pipelining	**5**	3	2	3	2	-	-	-	-	-	-	-	-	-	2	-	-

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				
PSO2	Design and develop secure, parallel, distributed, networked, and digital systems				
PSO3	Apply software engineering methods to design, develop, test and manage software systems.				
PSO4	Develop intelligent applications for business and industry				