1. The instruction, Add #45,R1 does \_\_\_\_\_\_\_

   **A.)** Adds the value of 45 to the address of R1 and stores 45 in that address

B.)   https://www.indianstudyhub.com/images/write.png    Adds 45 to the value of R1 and stores it in R1

   **C.)**Finds the memory location 45 and adds that content to that of R1

   **D.)**None of these

1. In the case of, Zero-address instruction method the operands are stored in \_\_\_\_\_\_\_\_.

   **A.)** Registers

   **B.)**Accumulators

C.)   https://www.indianstudyhub.com/images/write.png    Push down stack

   **D.)**Cache

1. Add #45, when this instruction is executed the following happen/s \_\_\_\_\_\_\_

   **A.)** The processor raises an error and requests for one more operand

B.)   https://www.indianstudyhub.com/images/write.png    The value stored in memory location 45 is retrieved and one more operand is requested

   **C.)**The value 45 gets added to the value on the stack and is pushed onto the stack

   **D.)**None of these

1. The addressing mode which makes use of in-direction pointers is \_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    Indirect addressing mode

   **B.)**Index addressing mode

   **C.)**Relative addressing mode

   **D.)**Offset addressing mode

1. In the following indexed addressing mode instruction, MOV 5(R1), LOC the effective address is \_\_\_\_\_\_

   **A.)** EA = 5+R1

   **B.)**EA = R1

   **C.)**EA = [R1]

D.)   https://www.indianstudyhub.com/images/write.png    EA = 5+[R1]

1. The addressing mode/s, which uses the PC instead of a general purpose register is \_\_\_\_\_\_\_\_.

   **A.)** Indexed with offset

B.)   https://www.indianstudyhub.com/images/write.png    Relative

   **C.)**Direct

   **D.)**Both Indexed with offset and direct

1. When we use auto increment or auto decrements, which of the following is/are true?  
   1) In both, the address is used to retrieve the operand and then the address gets altered  
   2) In auto increment, the operand is retrieved first and then the address altered  
   3) Both of them can be used on general purpose registers as well as memory locations

   **A.)** 1, 2, 3

   **B.)**2

   **C.)**1, 3

D.)   https://www.indianstudyhub.com/images/write.png    2, 3

1. The addressing mode, where you directly specify the operand value is \_\_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    Immediate

   **B.)**Direct

   **C.)**Definite

   **D.)**Relative

1. The effective address of the following instruction is MUL 5(R1,R2).

   **A.)** 5+R1+R2

   **B.)**5+(R1\*R2)

C.)   https://www.indianstudyhub.com/images/write.png    5+[R1]+[R2]

   **D.)**5\*([R1]+[R2])

1. \_\_\_\_\_ addressing mode is most suitable to change the normal sequence of execution of instructions.

A.)   https://www.indianstudyhub.com/images/write.png    Relative

   **B.)**Indirect

   **C.)**Index with Offset

   **D.)**Immediate

1. \_\_\_\_\_\_\_\_\_ converts the programs written in assembly language into machine instructions.

   **A.)** Machine compiler

   **B.)**Interpreter

C.)   https://www.indianstudyhub.com/images/write.png    Assembler

   **D.)**Converter

1. The instructions like MOV or ADD are called as \_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    OP-Code

   **B.)**Operators

   **C.)**Commands

   **D.)**None of these

1. The alternate way of writing the instruction, ADD #5,R1 is \_\_\_\_\_\_

   **A.)** ADD [5],[R1];

B.)   https://www.indianstudyhub.com/images/write.png    ADDI 5,R1;

   **C.)**ADDIME 5,[R1];

   **D.)**There is no other way

1. The assembler directive EQU, when used in the instruction: Sum EQU 200 does \_\_\_\_\_\_\_\_

   **A.)** Finds the first occurrence of Sum and assigns value 200 to it

B.)   https://www.indianstudyhub.com/images/write.png    Replaces every occurrence of Sum with 200

   **C.)**Re-assigns the address of Sum by adding 200 to its original address

   **D.)**Assigns 200 bytes of memory starting the location of Sum

1. The purpose of the ORIGIN directive is \_\_\_\_\_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    To indicate the starting position in memory, where the program block is to be stored

   **B.)**To indicate the starting of the computation code

   **C.)**To indicate the purpose of the code

   **D.)**To list the locations of all the registers used

1. The directive used to perform initialization before the execution of the code is \_\_\_\_\_\_

   **A.)** Reserve

   **B.)**Store

C.)   https://www.indianstudyhub.com/images/write.png    Dataword

   **D.)**EQU

1. \_\_\_\_\_ directive is used to specify and assign the memory required for the block of code.

   **A.)** Allocate

   **B.)**Assign

   **C.)**Set

D.)   https://www.indianstudyhub.com/images/write.png    Reserve

1. \_\_\_\_\_ directive specifies the end of execution of a program.

   **A.)** End

B.)   https://www.indianstudyhub.com/images/write.png    Return

   **C.)**Stop

   **D.)**Terminate

1. The last statement of the source program should be \_\_\_\_\_\_\_

   **A.)** Stop

   **B.)**Return

   **C.)**OP

D.)   https://www.indianstudyhub.com/images/write.png    End

1. When dealing with the branching code the assembler \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Replaces the target with its address

   **B.)**Does not replace until the test condition is satisfied

C.)   https://www.indianstudyhub.com/images/write.png    Finds the Branch offset and replaces the Branch target with it

   **D.)**Replaces the target with the value specified by the DATAWORD directive

1. The assembler stores all the names and their corresponding values in \_\_\_\_\_\_

   **A.)** Special purpose Register

B.)   https://www.indianstudyhub.com/images/write.png    Symbol Table

   **C.)**Value map Set

   **D.)**None of these

1. The assembler stores the object code in \_\_\_\_\_\_

   **A.)** Main memory

   **B.)**Cache

   **C.)**RAM

D.)   https://www.indianstudyhub.com/images/write.png    Magnetic disk

1. The utility program used to bring the object code into memory for execution is \_\_\_\_\_\_\_\_.

A.)   https://www.indianstudyhub.com/images/write.png    Loader

   **B.)**Fetcher

   **C.)**Extractor

   **D.)**Linker

.

1. In memory-mapped I/O \_\_\_\_\_\_\_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    The I/O devices and the memory share the same address space

   **B.)**The I/O devices have a separate address space

   **C.)**The memory and I/O devices have an associated address space

   **D.)**A part of the memory is specifically set aside for the I/O operation

1. The usual BUS structure used to connect the I/O devices is \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Star BUS structure

   **B.)**Multiple BUS structure

C.)   https://www.indianstudyhub.com/images/write.png    Single BUS structure

   **D.)**Node to Node BUS structure

1. The advantage of I/O mapped devices to memory mapped is \_\_\_\_\_\_\_\_\_\_\_

   **A.)** The former offers faster transfer of data

   **B.)**The devices connected using I/O mapping have a bigger buffer space

C.)   https://www.indianstudyhub.com/images/write.png    The devices have to deal with fewer address lines

   **D.)**No advantage as such

1. The system is notified of a read or write operation by \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Appending an extra bit of the address

   **B.)**Enabling the read or write bits of the devices

   **C.)**Raising an appropriate interrupt signal

D.)   https://www.indianstudyhub.com/images/write.png    Sending a special signal along the BUS

1. To overcome the lag in the operating speeds of the I/O device and the processor we use \_\_\_\_\_\_\_\_\_\_\_

   **A.)** BUffer spaces

B.)   https://www.indianstudyhub.com/images/write.png    Status flags

   **C.)**Interrupt signals

   **D.)**Exceptions

1. The method of accessing the I/O devices by repeatedly checking the status flags is \_\_\_\_\_\_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    Program-controlled I/O

   **B.)**Memory-mapped I/O

   **C.)**I/O mapped

   **D.)**None of these

1. The method of synchronising the processor with the I/O device in which the device sends a signal when it is ready is?

   **A.)** Exceptions

   **B.)**Signal handling

C.)   https://www.indianstudyhub.com/images/write.png    Interrupts

   **D.)**DMA

1. The method which offers higher speeds of I/O transfers is \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Interrupts

   **B.)**Memory mapping

   **C.)**Program-controlled I/O

D.)   https://www.indianstudyhub.com/images/write.png    DMA

1. The process wherein the processor constantly checks the status flags is called as \_\_\_\_\_\_\_\_\_\_\_

A.)   https://www.indianstudyhub.com/images/write.png    Polling

   **B.)**Reviewing

   **C.)**Inspection

   **D.)**Echoing

1. To overcome the problems of the assembler in dealing with branching code we use \_\_\_\_\_\_\_\_.

   **A.)** Interpreter

   **B.)**Debugger

   **C.)**Op-Assembler

D.)   https://www.indianstudyhub.com/images/write.png    Two-pass assembler

1. The interrupt-request line is a part of the \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Data line

B.)   https://www.indianstudyhub.com/images/write.png    Control line

   **C.)**Address line

   **D.)**None of these

1. The return address from the interrupt-service routine is stored on the \_\_\_\_\_\_\_\_\_\_\_

   **A.)** System heap

   **B.)**Processor register

C.)   https://www.indianstudyhub.com/images/write.png    Processor stack

   **D.)**Memory

1. The signal sent to the device from the processor to the device after receiving an interrupt is \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Interrupt-acknowledge

   **B.)**Service signal

   **C.)**Permission signal

   **D.)**Return signal

1. When the process is returned after an interrupt service \_\_\_\_\_\_ should be loaded again.  
   i) Register contents  
   ii) Condition codes  
   iii) Stack contents  
   iv) Return addresses

   **A.)** i, iv

   **B.)**ii, iii and iv

   **C.)**iii, iv

D.)   https://www.indianstudyhub.com/images/write.png    i, ii

1. The time between the receiver of an interrupt and its service is \_\_\_\_\_\_\_\_\_\_\_\_\_.

   **A.)** Interrupt delay

B.)   https://www.indianstudyhub.com/images/write.png    Interrupt latency

   **C.)**Cycle time

   **D.)**Switching time

1. Interrupts form an important part of \_\_\_\_\_\_\_\_\_\_\_ systems.

   **A.)** Batch processing

   **B.)**Multitasking

C.)   https://www.indianstudyhub.com/images/write.png    Real-time processing

   **D.)**Multi-user

1. \_\_\_\_\_\_ type circuits are generally used for interrupt service lines.  
   i) open-collector  
   ii) open-drain  
   iii) XOR  
   iv) XNOR

A.)   https://www.indianstudyhub.com/images/write.png    i, ii

   **B.)**ii

   **C.)**ii, iii

   **D.)**ii, iv

1. An interrupt that can be temporarily ignored is \_\_\_\_\_\_\_\_\_\_\_

   **A.)** Vectored interrupt

   **B.)**Non-maskable interrupt

C.)   https://www.indianstudyhub.com/images/write.png    Maskable interrupt

   **D.)**High priority interrupt

1. The 8085 microprocessor responds to the presence of an interrupt \_\_\_\_\_\_\_\_\_\_\_

   **A.)** As soon as the trap pin becomes ‘LOW’

   **B.)**By checking the trap pin for ‘high’ status at the end of each instruction fetch

C.)   https://www.indianstudyhub.com/images/write.png    By checking the trap pin for ‘high’ status at the end of execution of each instruction

   **D.)**By checking the trap pin for ‘high’ status at regular intervals

1. CPU as two modes privileged and non-privileged. In order to change the mode from privileged to non-privileged.

   **A.)** A hardware interrupt is needed

B.)   https://www.indianstudyhub.com/images/write.png    A software interrupt is needed

   **C.)**Either hardware or software interrupt is needed

   **D.)**A non-privileged instruction (which does not generate an interrupt)is needed

1. Which interrupt is unmaskable?

   **A.)** RST 5.5

   **B.)**RST 7.5

C.)   https://www.indianstudyhub.com/images/write.png    TRAP

   **D.)**Both RST 5.5 and 7.5

1. From amongst the following given scenarios determine the right one to justify interrupt mode of data transfer.  
   i) Bulk transfer of several kilo-byte  
   ii) Moderately large data transfer of more than 1kb  
   iii) Short events like mouse action  
   iv) Keyboard inputs

   **A.)** i and ii

   **B.)**ii

   **C.)**i, ii and iv

D.)   https://www.indianstudyhub.com/images/write.png    iv

1. How can the processor ignore other interrupts when it is servicing one \_\_\_\_\_\_\_\_\_\_\_

   **A.)** By turning off the interrupt request line

   **B.)**By disabling the devices from sending the interrupts

   **C.)**BY using edge-triggered request lines

D.)   https://www.indianstudyhub.com/images/write.png    All of the mentioned