TOUCHPAD
Computer Textbook Series

11

Computer Science with PYTHON

Beta*
*Updated Copy coming soon

ORANGE

Class XI

Answer Key

Computer Science with PYTHON

1. Computer Fundamentals

Assessment

A.

1. a	2. b	3. b	4. a	5. c
6. d	7. a	8. b	9. c	10. ?

B.

1. False	2. True	3. True	4. True	5. False
6. True	7. False	8. True	9. False	

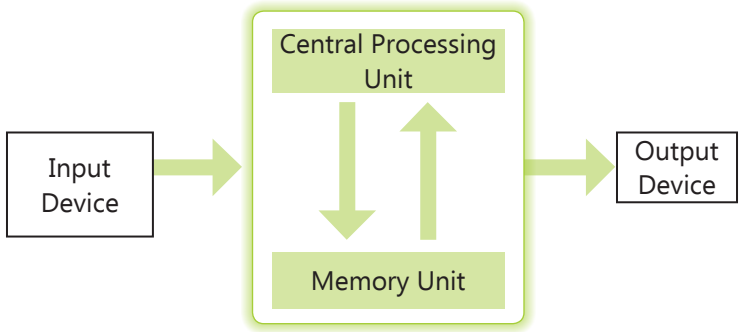
C.

1. System software	2. RAM, ROM	3. Utility	4. Bit
5. Byte	6. USB	7. Operating system	8. Biometric scanner

D.

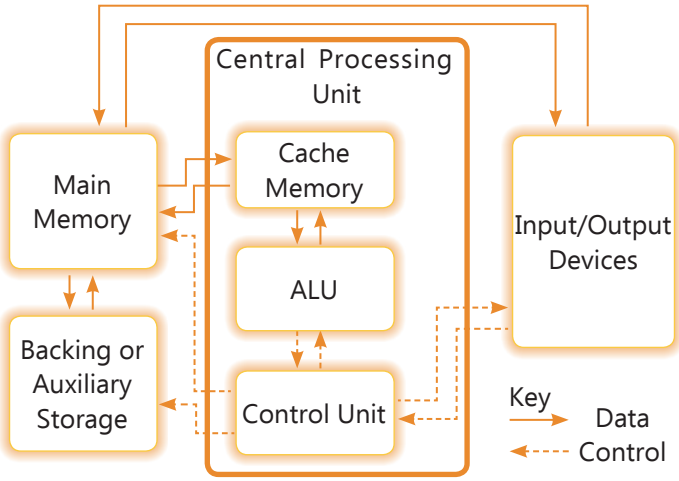
1. A computer is an electronic device that accepts input data from the user, processes it, and produces results. It comprises of the following components:
   a. Input Unit
   b. Central Processing Unit (CPU)
   c. Memory Unit
   d. Output Unit
2. Hardware and software are both essential for a computer to function. Hardware refers to the physical components of a computer, such as the CPU, memory, storage devices, and input/output devices. Software, on the other hand, consists of the programs and operating systems that run on the hardware. Without hardware, software cannot operate, and without software, hardware is useless. Together, they enable the computer to perform tasks such as processing data, executing applications, and managing resources.
3. Computer is an electronic device that accepts input data from the user, processes it, and produces results. It comprises of the following components:
   - Input Unit
   - Central Processing Unit (CPU)

- Memory Unit
- Output Unit



4. A light pen is a pointing device that allows users to interact with a computer screen by detecting light from the screen, whereas a mouse is a handheld pointing device that detects two-dimensional motion relative to a surface. A light pen is used directly on the display surface, while a mouse is used on a separate flat surface.
5. A barcode encodes information in the form of a sequence of lines of different lengths, heights and widths. A bar code reader fetches the information from a barcode printed on a product or a service by scanning its bar code. The sealed packets of any product at any shop typically includes a barcode containing information about the products and their prices. It is commonly used in retail stores, libraries, and warehouses to quickly retrieve product information, manage inventory, and streamline checkout processes.
6. The Central Processing Unit (CPU), also called the processor, is the computer's brain and is responsible for carrying out all the processing in the computer system. Given a task, the CPU is provided with a sequence of instructions in the form of a program.

   The CPU controls the entire data flow and instructions inside the computer. To facilitate functioning, the CPU comprises the following components:



- **Registers:** While processing an instruction, the CPU needs to store in its local storage the instruction, memory addresses being referred during the instruction execution, data operands (on which operation is to be carried out), and the result of the computation. For storing and accessing the information mentioned above, the CPU requires high-speed temporary storage units, called registers. Several registers are also equipped with circuitry for performing elementary operations like addition and subtraction. However, as the registers are much more expensive than the main memory, the CPU has only a limited number of them.
- **Arithmetic Logic Unit (ALU):** The arithmetic Logic Unit (ALU) of the CPU performs all arithmetic (+,-,*,/) and logical (>,<,>=,<=,<>) operations. While the result of an arithmetic operation is a numeric value, the result of a logical operation (such as 7<8) is either True or False. The values True and False are called Boolean values in honour of George Boole, who developed an algebra (again named, Boolean algebra, in his honour) that constitutes the basis of all computer computations. The data for executing an instruction may already be available in the registers or fetched in the ALU registers from the computer's memory. The result of a computation is often stored in the computer's memory.
- **Control Unit (CU):** The Control Unit (CU) controls the execution of instructions and the flow of data amongst the components of a computer, i.e., from input devices to memory, memory to ALU and vice versa, and from memory to output devices. It sends instructions in the form of control signals to ALU to perform the required arithmetic and/or logical operations.

7. Biometric sensors are used in:

   Security systems: To authenticate users based on their unique biological traits, such as fingerprint, retina, or facial recognition.

   Time and attendance systems: To track employee attendance and working hours using biometric data.

8. i. 1 GB = 2^{20}KB

   ii. 1 YB = 2^{50}GB

   iii. 1 KB = $2^{(-30)}$TB

   iv. 1 PB = 2^{30}MB

9. i. Primary memory and secondary memory:

   Primary memory (RAM) is fast, volatile memory used for storing data temporarily while the computer is running. Secondary memory (HDD, SSD) is slower, non-volatile memory used for long-term storage of data and programs.

   ii. System software and application software:

   System software includes the operating system and utility programs that manage computer resources and provide a platform for running application software. Application software includes programs designed to perform specific tasks for users, such as word processors, web browsers, and games.

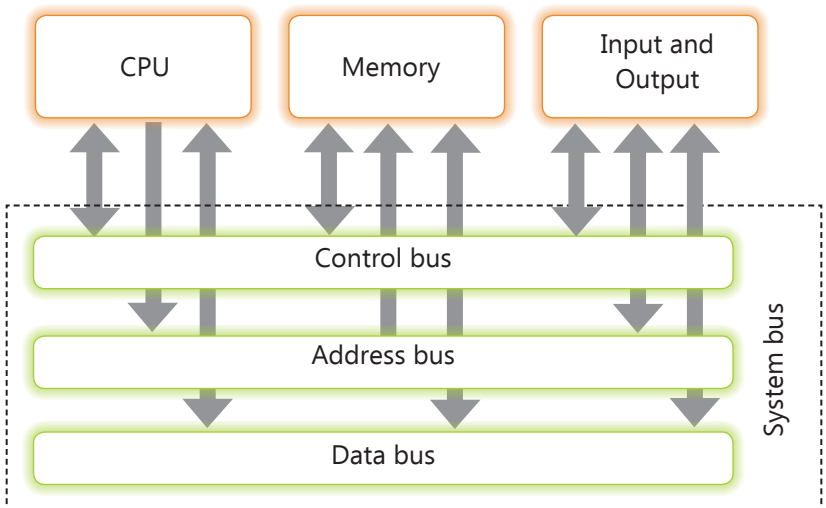
iii) Open source software and proprietary software:

Open source software is software with source code that anyone can inspect, modify, and enhance. Proprietary software is software that is owned by an individual or company and is distributed under a license that restricts modification and sharing.

10. Data is transferred between different components of a computer system (CPU to primary memory and vice versa or primary memory to secondary memory, or cache memory to CPU and vice versa) using physical connections called buses. There are three types of buses:

- **Data Bus:** It is used to transfer data between different CPU components.
- **Address Bus:** It is used to transfer addresses of memory locations for the CPU to perform read/write operations in the memory.
- **Control Bus:** It is used to transfer control signals between various computer components.

The data bus, address bus, and control bus together constitute the system bus. Figure 1.6 depicts the flow within the system bus. Data entered from an input device is stored in the main memory. The CPU then accesses this data using a data bus. The address of the memory location from where data will be accessed is communicated via the address bus. The read or write instructions are communicated through the control bus. Similarly, to write data into the memory, the CPU places the data on the data bus, and the address of the memory location where the data has to be written is placed on the address bus. Once this is done, a control signal is issued to write the data from the CPU to the designated memory location.



11. The functions of an operating system include:

- **Process management:** Manages the creation, scheduling, and termination of processes.
- **Memory management:** Allocates and deallocates memory space as needed by programs.
- **File system management:** Manages the creation, deletion, and access of files on storage devices.
- **Device management:** Controls and coordinates the use of hardware devices.

- **User interface:** Provides a way for users to interact with the computer, such as through a graphical user interface (GUI) or command-line interface (CLI).
- **Security and access control:** Protects data and resources from unauthorized access.

12. Three types of user interfaces are:
    - **Graphical User Interface (GUI):** Uses visual elements like windows, icons, and menus for interaction.
    - **Command-Line Interface (CLI):** Uses text-based commands for interaction.
    - **Touch User Interface (TUI):** Uses touch-based inputs for interaction, commonly found on smartphones and tablets.

Case-based Questions

1. a. Joey should use presentation software to create his presentation. An example of such software is Microsoft PowerPoint.

   b. The school is using customized application software. This is because the software has been specifically modified or developed to meet the specific needs and feedback of the students.

   c. There could be several reasons for this issue:
      - The printer driver may not be installed or may be outdated. Installing or updating the printer driver can solve this problem.
      - The printer may not be properly connected to the computer or network.
      - There could be a hardware issue with the printer or the computer's USB port.
      - The printer might be out of paper or ink, or there could be a paper jam.
      - The printer may not be set as the default printer in the computer's settings.

   d. **Interpreted Language:** In an interpreted language, the code is executed line-by-line by an interpreter at runtime. This means that the program can be run directly from the source code without the need for prior compilation. Examples include Python, JavaScript, and Ruby.

      Compiled Language: In a compiled language, the source code is first converted into machine code by a compiler. This machine code is then executed by the computer's hardware. The compilation step happens before the program is run, and the resulting machine code is what gets executed. Examples include C, C++, and Java.

2. a. The user interacts by typing commands using a Command-Line Interface (CLI).

   b. The interface that uses human touch to give instructions to the computer is a Touch User Interface (TUI).

   c. The interface that comprises menus and icons that are accessible by double-clicking is a Graphical User Interface (GUI).

2. Number Systems and Encoding Schemes

Assessment

A. 1. a 2. d 3. a 4. b 5. b
6. b 7. a 8. d 9. b 10. c

B. 1. False 2. False 3. True 4. True 5. False

C. 1. the most significant bit 2. 10 3. three 4. one
5. the English

D. 1. Hexadecimal numbers are often used in computer science and digital electronics to represent binary data in a more readable form. For example, memory addresses and color codes in web design use hexadecimal representation.

2. RGB codes are expressed as hexadecimal numbers because it is a concise way to represent large binary numbers, making it easier to read and understand color values.

3. UNICODE was developed to provide a universal character encoding standard that can represent text and symbols from all the writing systems in the world, ensuring consistent encoding, representation, and handling of text.

4. One advantage of UNICODE is that it supports a wide range of characters and symbols from different languages and scripts, enabling global interoperability and data exchange.

5. **ASCII:** American Standard Code for Information Interchange

   ISCII: Indian Script Code for Information Interchange

   UTF: Unicode Transformation Format

6. a. 25610=1000000002

   b. 39.56: 3910=1001112

   0.5610≈0.10001100110011001100110011002

   So, 39.5610≈100111.10001100110011001100110011002

   c. 508.8:

   50810=1111111002

   0.810≈0.11001100110011001100110011002

   So, 508.810≈111111100.11001100110011001100110011002

7. a. $5063_8 = 5 \times 8^3 + 0 \times 8^2 + 6 \times 8^1 + 3 \times 8^0$

   $= 5 \times 512 + 0 \times 64 + 6 \times 8 + 3 \times 1$

   $= 2560 + 48 + 3 = 2611_{10}$

b. 17_8 $=1 \times 8^1 + 7 \times 8^0 = 8 + 7 = 15_{10}$

$0.52_8 = 5 \times 8^{-1} + 2 \times 8^{-2} = 5 \times 0.125 + 2 \times 0.015625 = 0.625 + 0.03125 = 0.65625_{10}$

So, $17.52_8 = 15.65625_{10}$

c. 327:

$327_8 = 3 \times 8^2 + 2 \times 8^1 + 7 \times 8^0$

$= 3 \times 64 + 2 \times 8 + 7 \times 1$

$= 192 + 16 + 7 = 215_{10}$

8. a. 7208:

$7208_{10} = 1C2816$

b. 93.5:

$93_{10} = 5D_{16}$

$0.5_{10} = 0.8_{16}$

So, $93.5_{10} = 5D.8_{16}$

c. 199.60:

$199_{10} = C7_{16}$

$0.60_{10} \approx 0.9A_{16}$

So, $199.60_{10} \approx C7.9A_{16}$

9. a. 7C6:

$7C6_{16} = 7 \times 16^2 + C \times 16^1 + 6 \times 160$

$= 7 \times 256 + 12 \times 16 + 6 \times 1$

$= 1792 + 192 + 6 = 1990_{10}$

b. 95.A:

$95_{16} = 9 \times 16^1 + 5 \times 16^0 = 9 \times 16 + 5 \times 1 = 144 + 5 = 149_{10}$

$0.A_{16} = A \times 16–1 = 10 \times 0.0625 = 0.625_{10}$

So, $95.A_{16} = 149.625_{10}$

c. ABC.D:

$ABC16 = A \times 16^2 + B \times 16^1 + C \times 16^0$

$= 10 \times 256 + 11 \times 16 + 12 \times 1 = 2560 + 176 + 12 = 2748_{10}$

$0.D_{16} = D \times 16^{-1} = 13 \times 0.0625 = 0.812510$

So, $ABC.D_{16} = 2748.8125_{10}$

10. a. 1110001110:

$1110001110_2 = 38E_{16}$

b. 10101010:

$10101010_2 = AA_{16}$

c. 1100.01:

$1100.01_2 = C.2_{16}$

11. a. 245.8:

- Integer Part (245):

  $245_{10} = 11110101_2$

- Fractional Part (0.8):

  $0.8 \times 2 = 1.6 \rightarrow 1$

  $0.6 \times 2 = 1.2 \rightarrow 1$

  $0.2 \times 2 = 0.4 \rightarrow 0$

  $0.4 \times 2 = 0.8 \rightarrow 0$

  $0.8 \times 2 = 1.6 \rightarrow 1$

  This repeats as $0.8_{10} = 0.1100\overline{11}_2$

  So, $245.8_{10} \approx 11110101.1100\overline{11}_2$

b. 825:

$825_{10} = 1100111001_2$

c. 199.6:

- Integer Part (199):

  $199_{10} = 11000111_2$

- Fractional Part (0.6):

  $0.6 \times 2 = 1.2 \rightarrow 1$

  $0.2 \times 2 = 1.4 \rightarrow 0$

  $0.4 \times 2 = 0.8 \rightarrow 0$

  $0.8 \times 2 = 1.6 \rightarrow 1$

  This repeats as $0.6_{10} = 0.1001\overline{10}_2$

  So, $199.6_{10} \approx 11000111.1001\overline{10}_2$

12. a. BDA:

BDA16 = 10111101 10102

b. 5FA1:

$5F_{16} = 0101\ 1111_2$

$A1_{16} = 1010\ 0001_2$

So, $5F.A1_{16} = 0101\ 1111.1010\ 0001_2$

c. 892:

892_{16} = 10000 1001 0010_2

13. $10001002=68_{10}$='D'

$10000012=65_{10}$='A'

$10101002=84_{10}$='T'

$10001002=68_{10}$='D'

14. 'R' = 82_{10} = 01010010_2

'e' = 101_{10} = 01100101_2

'd' = 100_{10} = 01100100_2

So, Red in binary ASCII encoding is: '01010010 01100101 01100100'

Case-based Questions

1. Nirankar's computation has a mistake in the placement of powers of 16 and their multiplication. Here is the correct computation:

$(D20.B1)_{16}$

$= D \times 16^2 + 2 \times 16^1 + 0 \times 16^0 + B \times 16^{-1} + 1 \times 16^{-2}$

$= 13 \times 256 + 2 \times 16 + 0 \times 1 + 11 \times 0.0625 + 1 \times 0.00390625$

$= 3328 + 32 + 0 + 0.6875 + 0.00390625$

$= 3360.69140625_{10}$

So, the correct computation is $(3360.69140625)_{10}$

2. Sheikh's table for colors and decimal and hexadecimal values is as follows:

Name of Colour	Decimal Value	Hexadecimal Value
Yellow	(255, 255, 0)	(FF, FF, 00)
Red	(256, 0, 0)	(FF, 00, 00)
Blue	(0, 0, 255)	(00, 00, FF)
Black	(0, 0, 0)	(00, 00, 00)
Green	(0, 255, 0)	(00, FF, 00)

(256, (0(0, 255, 0)

3. Saptarishi's task to encode the words using ASCII and their binary equivalents is as follows:

Word	ASCII Code	Binary equivalent
BOT	66 79 84	01000010 01001111 01010100
KEY	75 68 89	01001011 01000100 01011001

NUMber	78 85 77 98 101 114	1001110 1010101 1001101 1100010 1100101 1110010
Yes–NO	89 101 115 45 78 79	01011001 01100101 01110011 00101101 010001110 01001111

So, here are the words with their ASCII codes and binary equivalent.

1. **BOT:**
   - ASCII Code: 66 79 84
   - Binary Equivalent: 01000010 01001111 01010100
2. **KEY:**
   - ASCII Code: 75 68 89
   - Binary Equivalent: 01001011 01000100 01011001
3. **NUMber:**
   - ASCII Code: 78 85 77 98 101 114
   - Binary Equivalent: 01001110 01010101 01001101 01100010 01100101 01110010
4. **Yes–NO:**
   - ASCII Code: 89 101 115 45 78 79
   - Binary Equivalent: 01011001 01100101 01110011 00101101 01001110 01001111

3. Boolean Logic

Assessment

A.	1.	c	2.	a	3.	b	4.	c	5. c
	6.	d	7.	d	8.	a	9.	b	10. a
B.	1.	False	2.	True	3.	False	4.	False	5. False
	6.	True	7.	False	8.	True	9.	False	
C.	1.	George Boole	2.	one	3.	binary	4.	1	5. NOR, NAND
	6.	Truth Table	7.	OR	8.	complement	9.	NAND	10. XNOR

D. 1. George Boole developed Boolean algebra, which provides a sound basis for studying logic circuits. Although one can define a Boolean algebra of 2, 4, 8, or more elements, in this book, we will restrict ourselves to the Boolean algebra of two elements as it serves as the basis of designing computer circuits.

Boolean algebra β is defined by a triplet β = < **S**, +, ●>, where **S** is a set having two elements 0 and 1.

1. Closure Property

(1) $a + b \in S, \forall a, b \in S$

(2) $a \bullet b \in S, \forall a, b \in S$

2. Commutative Property

(1) $a + b = b + a, \forall a, b \in S$

(2) $a \bullet b = b \bullet a, \forall a, b \in S$

3. Distributive Property

(1) $a + (b \bullet c) = (a + b) \bullet (a + c), \forall a, b, c \in S$

(2) $a \bullet (b + c) = a \bullet b + a \bullet c, \forall a, b, c \in S$

4. Identity Property

There exist two elements in **S**, denoted by 0 and 1, called identity of + and ● respectively, satisfying

(1) $a + 0 = a, \forall a \in S$

(2) $a \bullet 1 = a, \forall a \in S$

5. Complementarity Property

For each $a \in S$, there exists an element in S, denoted by a', such that

(1) $a + a' = 1$

(2) $a \bullet a' = 0$

a' is called the complement of a.

2. a. Associative Law:

(i) $(a + b) + c = a + (b + c)$

(ii) $(a \bullet b) \bullet c = a \bullet (b \bullet c)$

A	B	C	(A + B)	(A + B) + C	(B + C)	A + (B + C)
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	1	0	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

(ii) (A.B). C = A.(B.C)

A	B	C	A.B	(A.B).C	(B.C)	A.(B.C)
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1

0	1	1	1	1	1	1
1	0	0	1	1	0	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

b. Distributive Law

(i) A.(B+C) = (A.B) + (A.C)

A	B	C	(B + C)	A. (B + C)	A.B	A.C	A.B + A.C
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

(ii) A + B.C = (A+B) . (A+C)

A	B	C	BC	A + BC	A + B	A + C	(A+B). (A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

3. a.

A	B	C	A+C (LHS)	A	A'.C	A + A'.C	B.C	A + A'.C + B.C (RHS)
0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	0	0	1
0	1	0	0	1	0	0	0	0
0	1	1	1	1	1	1	1	1
1	0	0	1	0	0	1	0	1
1	0	1	1	0	0	1	0	1
1	1	0	1	0	0	1	0	1
1	1	1	1	0	0	1	1	1

b.

X	Y	Z	X + Z (LHS)	X′	X′.Z	X + X′Z	Y.Z	X + X′.Z + Y.Z (RHS)
0	0	0	0	1	0	0	0	0
0	0	1	1	1	1	1	0	1
0	1	0	0	1	0	0	0	0
0	1	1	1	1	1	1	1	1
1	0	0	1	0	0	1	0	1
1	0	1	1	0	0	1	0	1
1	1	0	1	0	0	1	0	1
1	1	1	1	0	0	1	1	1

4. a.

U	V	W	F(U, V, W)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0

b.

A	B	C	F(A, B, C)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

5. a. $F = (X^1 + Y) + (Y . Z^1)$

   b. $F = (P^1 + Q) + (Q . R^1)$

   c. $Y = (U . V^1 + U^1 . W^1)$

   d. $F = (P . Q^1) + (P^1 . R)$

6. De Morgan's laws: $\forall$ **a**, **b** $\in$ **S**

   (i) **(a + b)′ = a′ • b′**

(ii) $(a \bullet b)' = a' + b'$

Proof: (i)To prove $(a + b)' = a' \bullet b'$, we shall show that $a' \bullet b'$ is the complement of $(a + b)$

Find we prove: $(a + b) + (a' \bullet b') = 1$

$(a + b) + (a' \bullet b')$

$= a + (b + (a' \bullet b'))$ (Using associativity property)

$= a + ((b + a') \bullet (b + b'))$ (Using distributive property of +)

$= a + (b + a') \bullet 1$ (Using complementarity property)

$= a + (b + a')$ (Using multiplicative identity)

$= a + (a' + b)$ (Using commutative property)

$= (a + a') + b$ (Using associativity property)

$= 1 + b$ (Using complementarity property)

$= 1$ (Using Theorem 2 (i))

Next, we prove: $(a + b) \bullet (a' \bullet b') = 0$

$(a + b) \bullet (a' \bullet b')$

$= a \bullet (a' \bullet b') + b \bullet (a' \bullet b')$ (Using distributive property of $\bullet$)

$= (a \bullet a') \bullet b' + b \bullet (a' \bullet b')$ (Using associative property)

$= 0 \bullet b' + b \bullet (a' \bullet b')$ (Using complementarity property)

$= 0 + b \bullet (a' \bullet b')$ (Using theorem 2 (ii))

$= b \bullet (a' \bullet b')$ (Using additive identity)

$= b \bullet (b' \bullet a')$ (Using commutative property)

$= (b \bullet b') \bullet a'$ (Using associative property)

$= 0 \bullet a'$ (Using complementarity property)

$= 0$ (Using theorem 2 (ii))

7. a. (i) $a + a \bullet b = a$

   (i) $a \bullet (a + b) = a$

   b. (1) $a + (b \bullet c) = (a + b) \bullet (a + c), \forall a, b, c \in S$

   (2) $a \bullet (b + c) = a \bullet b + a \bullet c, \forall a, b, c \in S$

   c. $\forall a \in S, (a')' = a$

8. $(A . B)^1 = A^1 + B^1$

   DeMorgan's First Theorem Proof using Truth Table

A	B	A′	B′	A.B	(A.B)′	A′+B′
0	0	1	1	0	1	1
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	1	0	0

9. Do It Yourself

   b. 95.A:

   $95_{16} = 9 \times 16^1 + 5 \times 16^0 = 9 \times 16 + 5 \times 1 = 144 + 5 = 149_{10}$

10. a.

X	X^1	$X+X^1$
0	1	1
1	0	1

The given equation is true and verified

b.

X	X^1	$X+X^1$
0	1	0
1	0	0

The given equation is true and verified

c.

X	X + 1
0	1
1	1

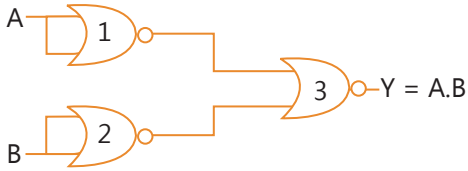
The given equation is true and verified

d.

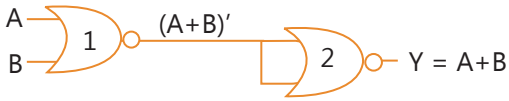
u	v	u^1	$u^1 + v$	$u. (u^1 + v)$	u + v
0	0	1	1	0	0
0	1	1	1	0	1
1	0	0	0	0	1
1	1	0	1	1	1

The given equation is FALSE.

11. a. AND gate using only NOR gate

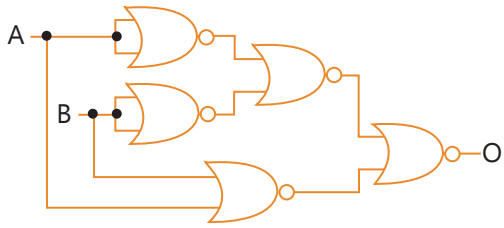


b. OR gate using only NOR gate

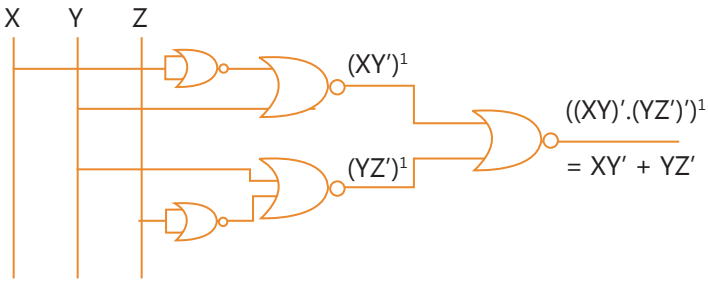


c. NOT gate using only NOR gate

d. XOR gate using only NOR gate



12. XY′ + YZ′



13. (i) A.(B+C) = (A.B) + (A.C)

A	B	C	B + C	A. (B + C)	A.B	A.C	A.B + A.C
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

(ii) A + B.C = (A+B) . (A+C)

A	B	C	BC	A+B+C	A+B	A+C	(A+B) . (A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

14. 2-input NAND gate

3-input NAND gate

15. 2-input NAND gate

X	**Y**	**Y^1**	**$X+Y^1$**	**$(X + Y^1)^1$**
0	0	1	1	0
0	1	0	0	1
1	0	1	1	0
1	1	0	1	0

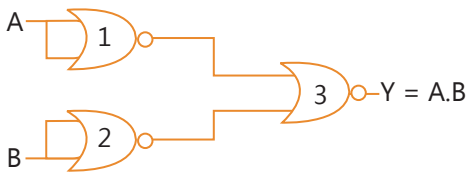
16.

A	B	C	A^1	AB	BC	CA^1	AB + BC + CA^1 (LHS)	AB + CA^1 (RHS)
0	0	0	1	0	0	0	0	0
0	0	1	1	0	0	1	1	1
0	1	0	1	0	0	0	0	0
0	1	1	1	0	1	1	1	1
1	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	1	0	0	1	0	0	1	1
1	1	1	0	1	1	0	1	1

The given expression is true and verified.

Case-based Questions

1. Implementing AND gate using NOR gate



2. Circuit -1: AND Gate

Truth Table:

A	B	(A • B) '
0	0	1
0	1	1
1	0	1
1	1	0

Circuit -1: AND Gate

Truth Table:

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

4. Problem Solving

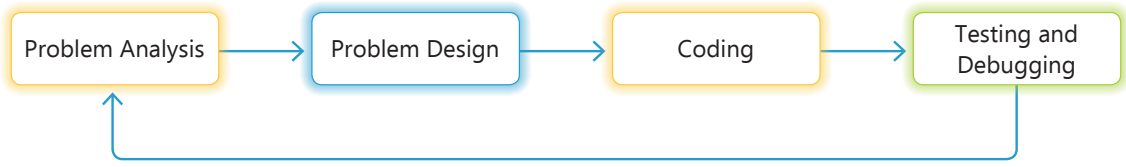
Assessment

A. 1. c 2. d 3. b 4. c
5. b 6. b

B. 1. True 2. False 3. False 4. True 5. True

C. 1. 1.flowchart, pseudocode 2. algorithm 3. pseudocode
4. testing and debugging

D. 1. It depicts the steps in problem solving using computers. The first step is to define a problem clearly. The second step is to analyze the problem and then develop a solution strategy (algorithm). Thereafter, in the coding phase, the code is written to implement the algorithm. Finally the code is tested for any type of errors in the Testing and Debugging phase.



2.
   - It requires some inputs from the user
   - It requires several steps to be executed
   - Each step of an algorithm must be executable
   - It produces some output
   - It should terminate after certain number of steps.
3. Input : Any two numbers (say N1, N2)

   Process: Compute the product

   Output: product of N1 and N2

```
input N1,  N2
product = N1 * N2
Print product
```

4. Input : Any two numbers (say N1, N2)

   Process: Check whether N1 is greater than N2

   Output: product

```
input N1,  N2
if N1 > N2 then
     Print N1
else if N2 > N1 then
     Print N2
else
     Print " Both are Equal"
```

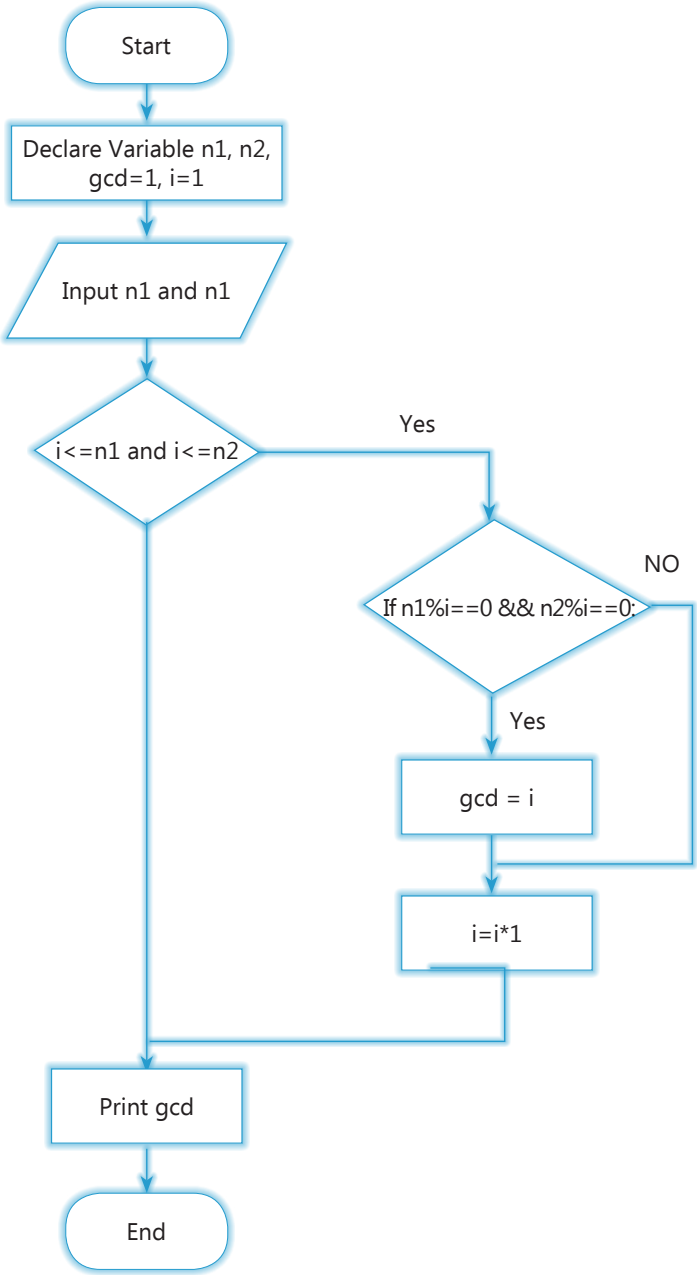
5. Input : Marks of four subjects - compSc, phy, chem, math

   Process: Computer aggregate and percentage of marks. Maximum Marks (say MM) is 100

   Output: Aggregate Marks (say aggr) and Percentage (say perc)

```
input compSc, phy, chem, math
aggr = compSc + phy + chem + math
perc = aggr / 400
Print aggr, perc
```

6.



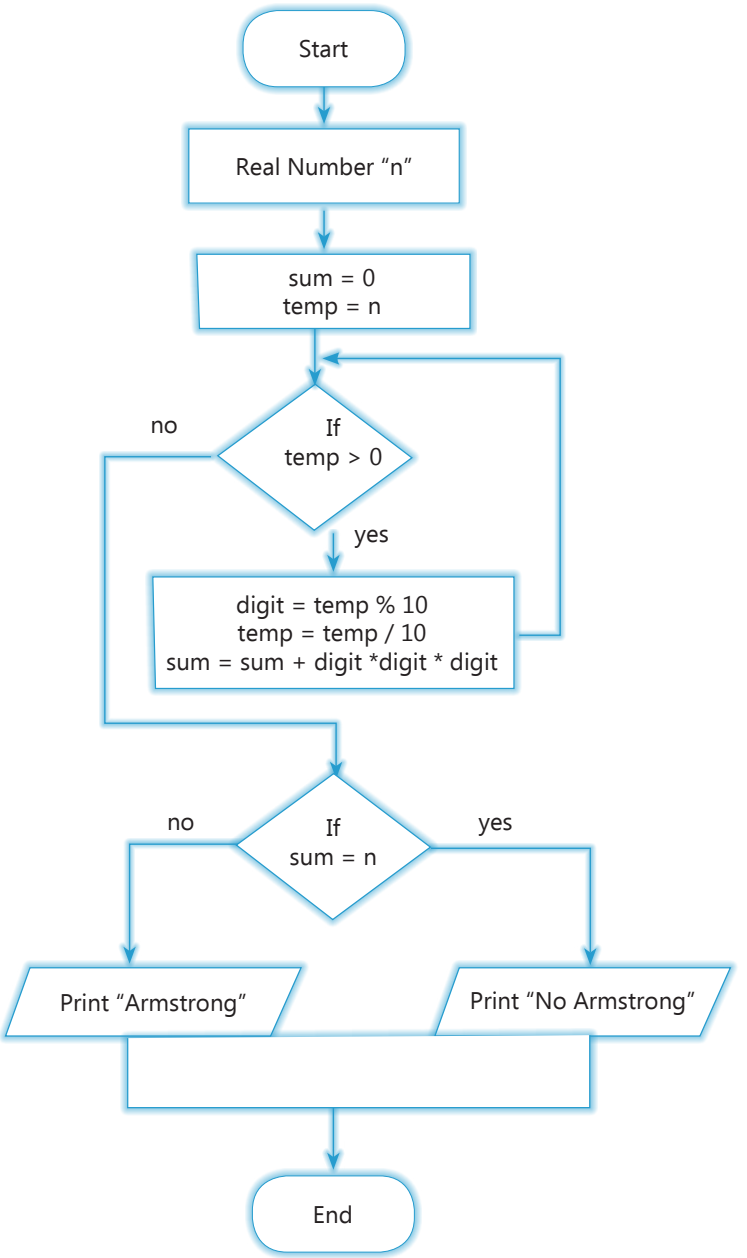
7. Input : Any number (say N1)

   Process: Check whether the number is a prime number or not

   Output: Display "PRIME NUMBER " if the number is prime, and "NOT PRIME" if it is not a prime number

8.

Flowchart - Armstrong Number

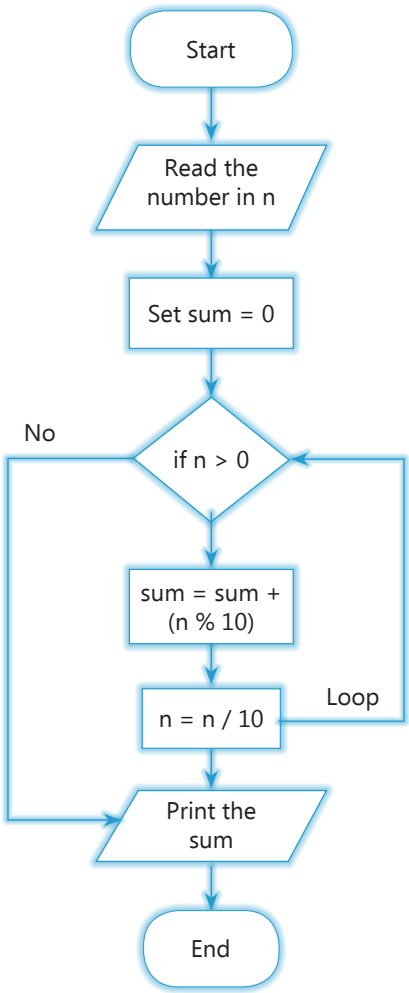


9. 1. `Start`
   2. `Sum = 0`
   3. `num = 1`
   4. `sum = sum + num`

5. num = num + 1
6. if num <= 10 then go to step 4 else print Sum
7. End

10. 1. Start
2. Input N1
3. digits = 0
4. num = N1 // 10
5. digits = digits + 1
6. if num !=0 then go to step 4 else display digits
7. End

11.



Flowchart to find sum of digits of a number

12. 1. b 2. a 3. c

13.

5. Getting Started With Python Programming

Assessment

A. 1. c 2. b 3. d 4. a 5. b

B. 1. True 2. False 3. False 4. False 5. True

C. 1. platform independent 2. Python script 3. .py
4. F5 5. print()

D. 1. Python is called an open source language because one can download the source code, modify it, and then redistribute it.

2. The various built-in functions within these libraries perform specialised tasks such as geosciences, life science, computational physics, etc. Generic libraries for machine learning and deep learning aid the development of several scientific and business applications.

3. It is easier to debug as it is an interpreter based language.

4. One advantage of UNICODE is that it supports a wide range of characters and symbols from different languages and scripts, enabling global interoperability and data exchange.

5. Python is considered a dynamically types language as it obviates the need to declare the types of variables.

6. This is because it is dynamically typed and interpreter based language.

7. Interactive Development Learning Environment

E. 1.
- Open any browser and type httpp://www.python.org/downloads/in the address bar.
- To download the desires Python version compatible with the operating system installed on your computer, click on the appropriate link. The click will initiate the download.
- Once the Python is downloaded, double-click on he executable file to run the setup. Follow the instructions to install the software.

2. print("ANOKHI")

3. a. 22.166666666666668

   b. 78.57142857142857

4. I am learning Python

   I am a class 11 student

   I will now play with numbers

150

25.142857142857142

Assertion and Reasoning Based Questions

1. b 2. a

Case-based Questions

1. Do It Yourself
2. Do It Yourself

6. Basics of Python Programming

Assessment

A.

1. b	2. a	3. c	4. b	5. a
6. c	7. b	8. d	9. a	10. b
11. d	12. a	13. c	14. c	15. b
16. c	17. a	18. b	19. a	20. c
21. c				

B.

1. True	2. False	3. False	4. True	5. False
6. False	7. True	8. True	9. False	10. True
11. True	12. False			

C.

1. keyword	2. underscore	3. delimitres	4. variable	5. assignment
6. newline	7. print()	8. int()		

D.

1. yes
2. operators, keywords, identifiers, literals, delimitres
3. `Asia&Europe - invalid ( use of & not allowed)`
   `Item-name - invalid ( use of - not allowed)`
   `while - invalid ( keyword)`
   `For - valid`
   `False - invalid (keyword)`
   `your_choice - valid`
   `input() - invalid ( use of ())`
   `output - valid`

4. Yes

   for eg : 10 + 20 will give output as 30 while "10" + "20" will give output as 1020

5. Yes. This is because a variable denotes or refres ot a data value

6. Rules for naming a variable are:

   - It should begin with an uppercase or lowercase alphabet, or an underscore (_).
   - The first letter of the variable (including an underscore) may be followed by an arbitrary combination of alphabets (a-z, A-Z), digits (0-9), and underscore (_).
   - A variable cannot start with a digit and cannot be a keyword.
   - The special symbols, such as !,@,#,$,%,^,&,*,(,), or blank spaces cannot be included in an identifier.
   - The identifiers are case-sensitive.

7. `weight = 15.6`

8. (i) =

   (ii) choice

   (iii) string

9. This is done by using assignment (=) after each variable name. For example to assign the value 0 to num1, num2, num3, and num4, the statement will be:

   ```
   num1 = num2 = num3 = num4 = 0
   ```

10. ```
    20
    20  -1
    ```

11. No. This is because the value of q is calculated but not displayed. The statement print(q) will produce the output.

12. `eighteighteight`

13. On execution of second statement, an error will be flashed as a string data cannot be divided.

14. Comments are statements that are ignored by the interpreter during the execution of the program. Unlike other statements in a program, comments have no effect on output of the program.

15. Using `sep` clause, the programmers can override the default separator space by specifying one of their choice.

    Using `end` clause, we can specify how to terminate the output displayed by `print()`

16. Single line comment:

    ```
    # Python program
    ```

    Multiline Comment

    ```
    ''' This is a multi\
    line comment '''
    ```
    ```

17. `print()`
18. A single line comment should begin with a hash (#)

    A multi line comment should be enclosed in triple quotes (''')
19. They are used to make the program more readable.

E. 1.
```
# Objective : To display name and age
name = input("Enter your name : ")
age = int(input("Enter your age : "))
print("Name : ", name)
print("Age : ", age)
```
2.
```
# Objective : To display city and state
city = input("Enter city : ")
state = input("Enter state : ")
print(city, state, sep = '\n')
```
3.
```
# Objective : Given the side of a cube, compute
# and display volume
side = int(input("Enter side of a cube : "))
volume = side * side * side
print("Volume of cube is : ", volume)
```
4.
```
# Objective : To display school name and address
school = input("Enter School Name : ")
address = input("Enter Address : ")
print("School Name : ", school)
print("Address : ", address)
```

Assertion and Reasoning Based Questions

1. c 2. b 3. a 4. c 5. a

Case-based Questions

1. Do It Yourself
2. Do It Yourself
3. Do It Yourself
4. Do It Yourself

7. Data Types and Operators

Assessment

A.

1.	a	2.	b	3.	None of the given options	4.	c	5.	d
6.	b	7.	b	8.	c	9.	c	10.	b

B.

1.	True	2.	True	3.	True	4.	True	5.	True
6.	True	7.	True	8.	True	9.	False	10.	False

C.

1.	type()	2.	False, True	3.	curly	4.	Identity	5.	ord()

D.

1. "kilogram" - string
   78.3 - float
   [90,45,23,76] - list
   ('a', 'b') - tuple
2. message[2] will give output 'u'
   message[5] will give output 'r'
3. Statement 1 is a list while statement 2 is a set.
4. This is because tuple is an immutable variable and hence cannot be changed using indexes.
5. Mutable : List, Dictionary
   Immutable: String, Tuple, Integer
6. String
7. A tuple is a sequence data type and is immutable.
   A dictionary is a mapping data type and is mutable.
8. String
9. Binary
10. (i) `5`
    (ii) `-64`
    (iii) `True`
11. (i) `*` is a product / multiplication operator. It multiplies two operands
    `**` is exponent operator.
    (ii) = is an assignment operator while == is a relational operator
    (iii) A relational operator is used to compare the values of operands on either side. A logical operator is yields True or False depending upon the logical operands on either side.

(iv) / is float division operator and yields the quotient as an integer value. // is an integer division operator and yields the quotient as float value.

12. An expression in Python is a valid combination of constants, variables, and operators.
13. An error will be generated as mathematical operations (difference) cannot be performed on a string. As int() is not used, the number entered by the user will be stored as a string value.
14.
```
# Objective : Accept a string(str) and a number(n) and
# and display the string n times
str = input("Enter a string : ")
n = int(input("Enter a number : "))
print(str * n)
```
15. A logical error does not give the desired output, even though there is no syntax error in the program. Such errors are sometimes difficult to identify as all the statements in the program are executed successfully.

    A runtime error occurs during the execution of the program. The statement is syntactically correct but cannot be executed by the interpreter.
16. 30.0
17. 22
18. No. 'None' is a string literal while None is a datatype that signifies absence of value.
19. False
20. True
21. (i) `False`

    (ii) `False`

    (iii) `False`

    (iv) `True`

    (v) `1.0`
22. `2.75 2 7 6.5 5`
23. It is difficult to locate logical errors as all the statements in the program are executed successfully.
24.
```
# Objective : To accept base length, breadth, and height of a parallelogram
# and display school its area and perimeter
baseLength = int(input("Enter a base length of a parallelogram : "))
breadth = int(input("Enter a breadth of a parallelogram : "))
height = int(input("Enter a height of a parallelogram : "))
area = baseLength * height
```

```
perimeter = 2 * ( breadth + height)
print("Area : ", area)
print("Perimeter : ", perimeter)
```

Assertion and Reasoning Based Questions

1. c 2. a 3. a 4. a 5. b

Case-based Questions

1.
```
# Objective : To evaluate an expression


a = 2
b = -1
c = 5
d = 3
value = (a + c) < b or d**a > 50
print("The value of given expression is ",value)
```

2.
```
# Objective : To accept a 2-digit number
# and display its reverse
num = int(input("Enter a two-digit number: "))
rem = num % 10 #remainder
quo = num // 10 #integers division
reversedNum = rem * 10 + quo
print("Reversed number:",reversedNum )
```

3.
```
# Objective : To accept time in seconds
# and display in minutes and seconds
secTime = int(input("Enter time in seconds: "))
timeMin = secTime // 60
timeSec = secTime % 60
print(timeMin,"minutes")
print(timeSec,"seconds")
```

4.
```
# Objective : To accept time in seconds
# and display in minutes and seconds
print("welcome to height calculator:)")
heightFeet = int(input("Enter height in feet: "))
```

```
heightInches = int(input("Enter remaining height in inches: "))
ti = heightFeet * 12 + heightInches
cm = ti * 2.54
print("YOUR HEIGHT IN CM IS:", cm, " cms")
```

8. Introduction to Functions

Assessment

A. 1. d 2. c 3. c 4. a 5. b
6. a 7. c 8. b 9. c

B. 1. True 2. False 3. True 4. True

C. 1. function 2. function name 3. first 4. function header
5. arguments

D. 1. The functions provide a systematic approach to problem-solving by dividing a complex problem into simpler sub-problems, developing solutions for each sub-problem, and combining the individual solutions of the sub-problems to solve the original problem.

2. The first line of function definition that ends with a colon is known as function header. It contains the function name and the list of dummy arguments or parameters that are given as input to the function.

3. a. **Parameters:** These are dummy arguments athat are given during function definition. These are the variables that receive values during the call to a function.

   Arguments: These are the input values passed as argument to the function.

   b. **Function definition:** Function definition begins with the keyword `def`, followed by the name of the function, followed by a pair of parenthesis that encloses a list of dummy arguments (if any) separated by commas. Dummy arguments are used as names of the objects for describing the computations inside a function

   Function Call: A function comes into action (i.e. gets executed) only when it is invoked. Invoking a function is called a function call.

4. no

5.
```
def square(num):
    '''#Objective : To compute and return square of a number
    Input Parameters : number - numeric value
    Return Value: square of the number - numeric value
    '''
```

```
    return (num * num)

number = int(input("Enter a number: "))
squareNum = square(number) # function call
print(squareNum)
```

6. You have an exam tomorrow
7. BIG*BIGGER*BIGGEST
   Jim Jam Jin##First-Sem-End#Festival#
8.
```
def compute(num1, num2):
    result=0
    result=num1 * num2
    return result
result = compute(10,20)
print("The result is ", result)
```

9. 1
10.
```
def checkOdd(num):
    '''Objective : To check for odd number
    Input Parameters : number - numeric value
    Return Value: True if number is odd,  Flase otherwise
    '''
    if num % 2 != 0:
        return True
    else:
        return False
```

11.
```
def calcPeri(radius):
    '''Objective : To compute and display perimeter of a circle
    Input Parameters : radius - numeric value
    Return Value: None
    '''
    perimeter = 2 * 3.14 * radius
    print("The perimeter is ", perimeter)
```

12.
```
def tempCelcius(tempF):
    '''Objective : To compute and return temperature in celcius
```

```
Input Parameters : temp in fahrenheit - numeric value
Return Value: temp in Celcius - numeric value
'''
tempC = (tempF - 32) * (5/9)
return tempC
```

Assertion and Reasoning Based Questions

1. a 2. c 3. b

Case-based Questions

1. Do It Yourself
2. Do It Yourself

9. Conditional Statements

Assessment

A. 1. b 2. c 3. a 4. b 5. c 6. d

B. 1. True 2. False 3. True 4. True 5. True

C. 1. sequential 2 .first 3. conditional 4. colon

D. 1. `pass` statement is used to leave a slot for the code to be filled in later. Sometimes, it is also used to simplify program logic when no action is required for a particular consition.

2. a. Simple Statements:
   - A simple statement appears by itself in a line.
   - The assignment statement, input and output statements are examples of simple statements.

   Compound Statements:
   - A compound statement often spans several lines. It comprises one or more header clauses and one or more sequences of statements, called suites.
   - A header clause begins with a keyword and ends with a colon.
   - A suite may comprise one or more statements and appears at the next level of indentation.

   b. Simple Statements:
   - A simple statement appears by itself in a line.

- The assignment statement, input and output statements are examples of simple statements.

`pass` Statements:

`pass` statement is used to leave a slot for the code to be filled in later. Sometimes, it is also used to simplify program logic when no action is required for a particular consition.

c. `if` statement:

- An if statement is the simplest compound statement that tests a condition.
- If the conditional expression yields True on its evaluation, the sequence of statements following the if header clause is executed.
- If the conditional expression yields False, the suite following the header is ignored by the interpreter.

`if-else` statement:

- If the conditional expression yields True on its evaluation, the sequence of statements following the if header clause is executed.
- If the conditional expression yields False, an else suite is executed.

3. Sometimes, there arises a need to check multiple conditional expressions. In such situations, we use the elif clause in an `if-else` statement.

4. a. `if name == "Gaurav"` or `marks > 50` and `marks < 90`:

   b. `if P > 78` or `Q <= 60`:

   c. `if city == "Delhi"` and `qualification == "Graduate"`:

5.
```
if num1 == num2 :
    print("Equal")
else:
    print("Unequal")
```

6.
```
if num1 < num2:
    if num2 < num3:
        print(num1 , num2, num3)
    else:
        if num1 < num3:
            print(num1, num3, num2)
        else:
            print(num3, num1, num2)
```

7. (i) (a) Multiple of 5

   (b) Not Multiple of 5

(ii) Success

(iii) (a) 66

(b) 16

(c) 8

(d) 17

(iv) 4

(v) Not Same

(vi) No output

(vii) TRUE

(viii) Hey

E. 1.
```
def smaller(num1,num2):
    '''
    Objective : To compare two numbers
    Input Parameters : num1 and num2 - numeric value
    Return Value: None
    '''
    if num1 < num2:
       print(num1, " is smaller")
    elif num2 < num1:
       print(num2, " is smaller")
    else:
       print("Both are equal")
number1 = int(input("Enter first number : "))
number2 = int(input("Enter second number : "))
smaller(number1, number2)
```

2.
```
def smallest(num1,num2,num3):
    '''
    Objective : To compare three numbers
    Input Parameters : num1, num2 and num3 - numeric value
    Return Value: None
    '''
    if num1 < num2 and num1 < num3:
```

```
        print(num1, " is smallest")
    elif num2 < num1 and num2 < num3:
        print(num2, " is smallest")
    elif num3 < num1 and num3 < num2:
        print(num3, "is smallest")
    elif num1 == num2 and num1 == num3:
    print("All three are equal")
number1 = int(input("Enter first number : "))
number2 = int(input("Enter second number : "))
number3 = int(input("Enter third number : "))
smallest(number1, number2, number3)
```

3.
```
def smallest(num1,num2,num3,num4):
    '''
    Objective : To compare four numbers
    Input Parameters : num1, num2, num3 and num4 - numeric value
    Return Value: None
    '''
    if num1 < num2 and num1 < num3 and num1 < num4:
        print(num1, " is smallest")
    elif num2 < num1 and num2 < num3 and num2 < num4:
        print(num2, " is smallest")
    elif num3 < num1 and num3 < num2 and num3 < num4:
        print(num3, "is smallest")
    elif num4 < num1 and num4 < num2 and num4 < num3:
        print(num4, "is smallest")
    elif num1 == num2 and num1 == num3 and num1 == num4:
        print("All four are equal")
number1 = int(input("Enter first number : "))
number2 = int(input("Enter second number : "))
number3 = int(input("Enter third number : "))
number4 = int(input("Enter fourth number : "))
smallest(number1, number2, number3, number4)
```

4.
```
def descOrder(num1,num2,num3):

    #Objective : To arrange numbers in descending order
    #Input Parameters : num1, num2, num3 - numeric values
    #Return Value: None

    if num1 > num2 and num1 > num3:
    if num2 > num3:
        print(num1,num2, num3)
    else:
        print(num1, num3, num2)
    elif num2 > num1 and num2 > num3:
    if num1 > num3:
        print(num2, num1, num3)
    else:
        print(num2, num3, num1)
    elif num3 > num1 and num3 > num1:
    if num1 >= num2:
        print(num3,num1, num2)
    else:
        print(num3, num2, num1)
number1 = int(input("Enter first number : "))
number2 = int(input("Enter second number : "))
number3 = int(input("Enter third number : "))
descOrder(number1, number2, number3)
```
5.
```
def checkMultiple(num1,num2):

    #Objective : To check whether whether first number is multiple of
second
    #Input Parameters : num1, num2 - numeric value
    #Return Value: None

    if num1 % num2 ==0:
```

```
            print(num1, " is multiple of ", num2)
        else:
            print(num1, " is not a multiple of ", num2)
    number1 = int(input("Enter first number : "))
    number2 = int(input("Enter second number : "))
    checkMultiple(number1, number2)
```

6.
```
    import math
    def findRoots(num1, num2, num3):
        #Objective : To find roots of a quadratic equation
        #Input Parameters : num1, num2, num3 - numeric value
        #Return Value: None

        discriminant = (num2 ** 2) - 4 * num1 * num3

        # Calculate two solutions
        root1 = (-num2 + math.sqrt(discriminant)) / (2*num1)
        root2 = (-num2 - math.sqrt(discriminant)) / (2*num1)

            print("The roots are : ", root1," and ", root2)
    number1 = int(input("Enter first number : "))
    number2 = int(input("Enter second number : "))
    number3 = int(input("Enter third number : "))
    findRoots(number1, number2, number3)
```

7.
```
    def areaPerimeter(side, choice):
        #Objective : To compute area and perimeter of a square
        #Input Parameters : side - numeric value, choice - character value
        #Return Value: None

        if choice == 'A':
           print("Area of square is : ", side * side)
        elif choice == 'P':
           print("perimeter of square is : ", 4 * side)
        else:
```

```
        print("Wrong choice ")

s = int(input("Enter side of a square : "))
ch = input("Enter A to view Area OR P to view Perimeter : ")
areaPerimeter(s, ch)
```

8.
```
import math
def difference(num1, num2):
    #Objective : To compute and display positive differecne between
two numbers
    #Input Parameters : num1, num2 - numeric values
    #Return Value: None

    diff = num1 - num2
       print("Postive difference is : ", math.fabs(diff))

number1 = int(input("Enter first number : "))
number2 = int(input("Enter second number : "))
difference(number1, number2)
```

9.
```
def calcGrades(name, marks, MM):
    '''
    Objective : To display grade
    Input Parameters : name - string marks, MM - numeric values
    Return Value: grade - string
    '''
    perc = (marks/MM) * 100
    if perc >= 90:
        grade = 'A'
    elif perc >= 75 and perc < 90 :
        grade = 'B'
    elif perc >= 60 and perc < 75 :
        grade = 'C'
    elif perc >= 40 and perc < 60 :
        grade = 'D'
```

```
    elif perc >= 33 and perc < 40:
        grade = 'E'
    elif perc < 33:
        grade = 'F'
    return grade

def acceptData():
    '''
    Objective : To accept name, marks and maximum marks
    Input Parameters : None
    Return Value: None
    '''
    sName = input("Enter your name : ")
    sMarks = int(input("Enter your marks : "))
    maxMarks = int(input("Enter Maximum Marks in examination : "))
    sGrade = calcGrades(sName, sMarks, maxMarks)
    print("Your Grade is : ", sGrade)
acceptData()
```

10.
```
def checkChar(ch):
   #Objective : To check and display whether a character is an alphabet/
digit/special character
    #Input Parameters : ch - string
    #Return Value: None

    if ch >= 'A' and ch <= 'Z':
        print(ch, " is an uppercase character.")
    elif ch >= 'a' and ch <= 'z':
        print(ch, " is a lowercase character.")
    elif ch >= '0' and ch <= '9':
        print(ch, " is a digit.")
    else:
        print(ch, " is a special character")
character = input("Enter a character : ")
checkChar(character)
```

11.
```
def calcBMI(weight, height):
    #Objective : To compute and display BMI
    #Input Parameters : weight, height - numeric values
    #Return Value: None

    bmi = weight / (height * height)
    print(" Your BMI is : ", bmi)

wt = float(input("Enter your weight in kg : "))
ht = float(input("Enter your height in metres : "))
calcBMI(wt, ht)
```

Assertion and Reasoning Based Questions

1. d

Case-based Questions

1. Do It Yourself
2. Do It Yourself

10. Looping in Python

Assessment

A.

1. a	2. a	3. c	4. a	5. a
6. c	7. d	8. c	9. d	

B.

1. False	2. True	3. False	4. False	5. True
6. True				

C.

1. iteration / loop	2. else	3. break	4. inner
5. iteration			

D. 1. a. **`for`** loop

- executes the statements in the body of the loop for finite number of time
- Syntax:

```
for control_variable in sequence / values in range:
```

```
    body of for loop
[else:
    statements]
```

while loop

- executes the statements in the body of the loop till the test condition is true
- Syntax:

```
while test condition:
    Body of while loop
[else:
    Statements]
```

b. **break** statement

- The `break` statement terminates the same loop in which it is defined and moves the control to the next statement immediately following the loop.
- Once the break statement is encountered and executed, no further statement in the loop will be executed.

continue statement

- When a `continue` statement is encountered, the remaining statement(s) in the loop is skipped and the control jumps to the beginning of the loop for the next iteration.

2. `else` clause in an `if` statement is executed if the test condition yields a False.

   `else` clause in a `for` loop is executed after all possible iterations of `for` loop are executed.

3. An infinite loop is a loop that never terminates. This is because the test conditions never yields False.

   For example:

```
while True:
    s = 1
    print(s)
```

   In the given example, the value of s will always be 1. Hence the while condition will never yield a False, resulting in an infinite loop.

4. The given code will result in an infinite loop. This is because the value of c will never become 0 and so the condition will never yield a False.

5.
```
for moon in range(50, 0, -5):
    print(moon // 5)
```

   I would prefer for loop, as the program is more readable and easy to understand.

6.
```
sun = 0
while sun < 5:
    moon = 0
    while moon < 3:
        if sun ** moon >= 5:
            print(sun)
        else:
            print(moon)
        moon = moon + 1
    sun = sun + 1
```

I would prefer for loop, as the program is more readable and easy to understand.

7. (i)
```
5#2
4#3
```
(ii) W o r l d P e a c e

(iii) 5
4
3

8. It is often used as a stub when we want to leave some functionality to be defined in the future. For example, let us assume you don't want to say how much a test result should be changed for moderation. Instead, you want it to be determined statistically from the result of the examination. However, you may not yet have thought about how to do it. So, you may define a function that just contains a pass statement. For example,

```
def moderate(result):
    pass
```

E. 1.
```
def sumNatural(n):
    #Objective : To display sum of first n natural numbers
    #Input Parameters : n - numeric value
    #Return Value: None

    sum = 0
    for x in range(n+1):
        sum = sum + x
    print("The sum of first ", n, " natural numbers is ", sum)
```

```
num = int(input("Enter a natural number :  "))
sumNatural(num)
```

2.
```
def sumOdd():
     #Objective : To accept 10 numbers and display sum of only odd numbers
    #Input Parameters : None
    #Return Value: None

    sum = 0
    for x in range(10):
        num = int(input("Enter a number"))
        if num % 2 != 0:
            sum = sum + num
    print("The sum is :  ", sum)
sumOdd()
```

3.
```
def secondLarge(n):
    #Objective : To accept n numbers and display the second largest number
    #Input Parameters : n - numeric value
    #Return Value: None
    num1 = int(input("Enter number 1 : "))
    maxNumber = secondMax = num1
    for x in range(n - 1):
        num = int(input("Enter a number :  "))
        if num > maxNumber:
            secondMax = maxNumber
            maxNumber = num
        elif maxNumber > num > secondMax:
            secondMax = num
    print("The second largest number is : ", secondMax)
totalNum = int(input("Enter the total number "))
secondLarge(totalNum)
```

4.
```
def fibonacci(n):
    #Objective : To display first n elements of Fibonacci series
    #Input Parameters : n - numeric value
    #Return Value: None
    a = 0
    b = 1
    print(a, b, sep = '  ', end = '  ')
    for x in range(n - 2):
        z = a + b
        print(z, end = ' ')
        a = b
        b = z
numElements = int(input("Enter the number of elements of a Fibonacci
series : "))
fibonacci(numElements)
```

5.
```
def isPrime(num):
    #Objective : To check whether a number is a prime number
    #Input Parameters : num - numeric value
    #Return Value: True or False - Boolean value

    if num <= 1:
        return False
    if num <= 3:
        return True
    if num % 2 == 0 or num % 3 == 0:
        return False
    i = 5
    while i * i <= num:
        if num % i == 0 or num % (i + 2) == 0:
            return False
        i += 6
    return True
```

```
def primeNumbers(num1, num2):
    #Objective : To display prime numbers between two numbers
    #Input Parameters : num1, num2 - numeric values
    #Return Value: None
    print("The Prime Numbers between ", num1, " and ", num2, "are : ")
    for x in range(num1, num2 + 1):
        if isPrime(x):
            print(x, end = '  ')

number1 = int(input("Enter first number :  "))
number2 = int(input("Enter second number :  "))
primeNumbers(number1, number2)
```

6.
```
def isPalin(num):
    #Objective : To check whether a number is a palindrome
    #Input Parameters : num - numeric value
    #Return Value: None

    actualNum = num
    reverseNum = 0
    while num > 0:
        rem = num % 10
        reverseNum = reverseNum * 10 + rem
        num //= 10
    if actualNum == reverseNum:
        print("PALINDROME")
    else:
        print("NOT a PALINDROME")

number = int(input("Enter a positive number :  "))
if number < 0:
    print("ONLY Positive number is to be entered ")
else:
    isPalin(number)
```

7.
```
def marksAnalysis():
    #Objective : To display average marks, student with highest and lowest marks
    #Input Parameters : None
    #Return Value: None

    sName = input("Enter your name :  ")
    sMarks = int(input("Enter your marks :  "))
    sumMarks = sMarks
    highestMarks = sMarks
    lowestMarks = sMarks
    for x in range(9):
        sName = input("Enter your name :  ")
        sMarks = int(input("Enter your marks :  "))
        sumMarks = sumMarks + sMarks
        if sMarks > highestMarks:
            highStudent = sName
        elif sMarks < lowestMarks:
            lowStudent = sName
    average = sumMarks / 10
    print("Average Marks : ", average)
    print("The student with highest marks : ", highStudent)
    print("The student with lowest marks : ", lowStudent)

marksAnalysis()
```

8.
```
def sumOfSeries(p, n):
    #Objective : To display sum of series 1-p+p2-p3+p4- ...pn
    #Input Parameters : p, n - numeric values
    #Return Value: sum of given series

    sum = 0
    sign = 1
    for x in range(n + 1):
```

```
            sum += sign * (p ** x)
            sign *= -1
        return sum
valueP = int(input("Enter the value of p :  "))
valueN = int(input("Enter the value of n :  "))
seriesSum = sumOfSeries(valueP, valueN)
print("The sum of given series is : ", seriesSum)
```

9.
```
def perfectNumber(num):
    #Objective : To check whether a number is a perfect number
    #Input Parameters : num - numeric value
    #Return Value: None

    if num <= 0:
        print("Only positive numbers can be checked for a perfect
        number")
    divisorsSum = 0
    for i in range(1, num):
        if num % i == 0:
            divisorsSum += i
    if divisorsSum == num:
        print(num, " is a PERFECT NUMBER")
    else:
        print(num, " is NOT a PERFECT NUMBER")

number = int(input("Enter a number  :  "))
perfectNumber(number)
```

10. i.
```
def leftTriangleAlpha():
    #Objective : To display left triangle of alphabets
    #Input Parameters : None
    #Return Value: None
    size = 5
    alpha = 65
    for i in range(size):
```

```
            for j in range(i+1):
                print(chr(alpha+j), end="")
            print()

    leftTriangleAlpha()
```

ii. Do It Yourself

(INverted Pyramid - program 10.3)

iii.

```
    def halfDumble():
        #Objective : To display half dumble pattern of stars
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))
        m=n

        while n>=1:
            print('*'*n)
            n-=1
        n+=2
        while n<=m:
            print('*'*n)
            n+=1

    halfDumble()
```

iv.

```
    def rightPascalTriangle():
        #Objective : To display right pascal triangle
        #Input Parameters : None
        #Return Value: None

        n = 5
        # upper triangle
        for i in range(n):
            for j in range(i + 1):
```

```
                print('*', end="")
            print()
        # lower triangle
        for i in range(n):
            for j in range(n - i - 1):
                print('*', end="")
            print()

    rightPascalTriangle()
```

v.

```
    def SquareOneDiag():
        #Objective : To display square with diagonal
        #Input Parameters : None
        #Return Value: None

        n=6
        for i in range(n):
            for j in range(n):
                if i == j:
                    print(i + 1, end=' ')
                else:
                    print(0, end=' ')
            print()
    SquareOneDiag()

    ##################double spaces
```

vi.

```
    def leftTriangleNum():
            #Objective : To display left triangle of numbers
            #Input Parameters : None
            #Return Value: None

            size = 5
            for i in range(size):
                for j in range(i+1):
```

```
                        print(j+1, end="  ")
                    print()
            leftTriangleNum()
            vii.def invertTriangleNum():
        #Objective : To display inverted triangle of numbers
        #Input Parameters : None
        #Return Value: None

        n=5
        for i in range(1,n+1):
            for j in range(n,i-1,-1):
                print(i, end='   ')
            print()
    invertTriangleNum()
viii.   def leftNumberTriangle():
        #Objective : To display left triangle of numbers
        #Input Parameters : None
        #Return Value: None

        size = 5
        for i in range(size):
            for j in range(i+1):
                print(j+1, end="  ")
            print()
    leftNumberTriangle()
ix.     def leftNumReverseTriangle():
        #Objective : To display left triangle of numbers with each
        line in reverse
        #Input Parameters : None
        #Return Value: None

        size = 5
        for i in range(1,size+1):
```

```
            for j in range(i, 0, -1):
                print(j, end="  ")
            print()
    leftNumReverseTriangle()
```

x.

```
    def numberSeriesTriangle():
        #Objective : To display triangle with numbers in series
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))

        a=0
        t=0
        for x in range(0,n):
            for y in range(x,0,-1):
                if y==1:
                    print(a+y)
                    t=1
                else:
                    print(a+y,end='  ')
            a+=x
    numberSeriesTriangle()
```

xi.

```
    def palindromeInvertedTriangle():
        #Objective : To display inverted triangle with each row as
        palindrome
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))
        print('Given a digit say,',n)
        m=n

        for x in range(0,n):
```

```
            for y in range(0,m-x):
                print(m-y,end='  ')
            for z in range(m-x,0,-1):
                if z==1:
                    print(m-z+1)
                else:
                    print(m-z+1,end='  ')

    palindromeInvertedTriangle()
```

xii.
```
    def leftReverseEvenTriangle():
        #Objective : To display reverse triangle with all even
        numbers
        #Input Parameters : None
        #Return Value: None

        n=5
        for i in range(1, n + 1):
            for j in range(i,i,-2):
                print(i, end=' ')
            print()
        print()
    leftReverseEvenTriangle()
```

xiii.
```
    def multiplesTriangle():
        #Objective : To display triangle of numbers with row of
        multiples
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))

        for x in range(0,n+1):
            if x==0:
                print(0)
```

```
                    continue
                for y in range(0,x+1):
                    if y==0:
                        print(0,end=' ')
                    elif y==x:
                        print(x*x)
                    else:
                        print(x*y,end=' ')

        multiplesTriangle()
```

xiv.

```
        def evenNumbersTraingle():
            #Objective : To display triangle with even numbers
            #Input Parameters : None
            #Return Value: None

            n=int(input('Enter number: '))
            if n%2==0:
                m=n-1
            else:
                m=n

            t=1
            for x in range(1,m+1,2):
                for y in range(0,t):
                    if y==t-1:
                        print(x)
                    else:
                        print(x,end='  ')
                t+=1

        evenNumbersTraingle()
```

xv.

```
        def squareFull():
            #Objective : To display full square
```

```
        #Input Parameters : None
        #Return Value: None

        size = 5
        for i in range(0, size):
            for j in range(0, size):
                print("*", end="  ")
            print()
```

xvi.
```
    def filledSquareStarsDiagCaret():
        #Objective : To display filled square of stars with left
        diagonal of carets
        #Input Parameters : None
        #Return Value: None


        n=int(input('Enter number: '))
        str1='* '*n

        for x in range(0,n+1):
            print(str1[0:2*x]+'^ '+str1[0:2*n-2*x])

    filledSquareStarsDiagCaret()
```

xvii.
```
    def squareStarLeftDiagCaret():
        #Objective : To display hollow square of stars with left
        diagonal of carets
        #Input Parameters : None
        #Return Value: None
        n=int(input('Enter number: '))

        for x in range(1,n+1):
            if x==1 or x==n:
                print('* '*n)
                continue
```

```
            print('* '+' '*(2*(x-2))+'^'+' '*(2*(n-x-1))+' *')
    squareStarLeftDiagCaret()
```

xviii.
```
    def sqaureStarsRightDiagCaret():
        #Objective : To display filled square of stars with right
        diagonal of carets
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))

        for x in range(1,n+1):
            if x==1 or x==n:
                print('* '*n)
                continue
            print('* '+'* '*(n-x-1)+'^ '+'* '*(x-2)+'*')

    sqaureStarsRightDiagCaret()
```

xix.
```
    def squareStarCaretDiagonal():
        #Objective : To display hollow square of stars with right
        diagonal of carets
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))

        for x in range(1,n+1):
            if x==1 or x==n:
                print('* '*n)
                continue
            print('* '+' '*(2*(n-x-1))+'^'+' '*(2*(x-2))+' *')


    squareStarCaretDiagonal()
```

xx. Do It Yourself

```
#Hollow star square with two caret diagonals
```

xxi.

```
def rightTriangleDots():
    #Objective : To display right triangle of dots
    #Input Parameters : None
    #Return Value: None
    n=int(input('Enter number: '))
    for x in range(1,n+1):
        print('.  '*x)
rightTriangleDots()
```

xxii.

```
def dotTraingleCaretHypt():
    #Objective : To display right triangle of dots with hypotenuse
    of carets
    #Input Parameters : None
    #Return Value: None

    n=int(input('Enter number: '))
    for x in range(0,n):
        print('* '*x+'^')

dotTraingleCaretHypt()
```

xxiii.

```
def TriangleCaretDots():
    #Objective : To display filled triagle of dots with sides
    in carets
    #Input Parameters : None
    #Return Value: None

    n=int(input('Enter number: '))
    for x in range(0,n):
        if x==0:
            print('^')
        elif x==n-1:
            print('^ '*n)
```

```
                else:
                    print('^ '+'. '*(x-1)+'^')

        TriangleCaretDots()
xxix.   def hollowTriangleCaret():
            #Objective : To display hollow triangle of carets
            #Input Parameters : None
            #Return Value: None

            n=int(input('Enter number: '))
            for x in range(0,n):
                if x==0:
                    print('^')
                elif x==n-1:
                    print('^ '*n)
                else:
                    print('^ '+'  '*(x-1)+'^')

        hollowTriangleCaret()
xxv.    def rightSideTraingleStars():
            #Objective : To display right sided triangle with stars
            #Input Parameters : None
            #Return Value: None

            n=int(input('Enter number: '))
            for x in range(1,n+1):
                print('  '*(n-x)+'* '*x)
        rightSideTraingleStars()
xxvi.   def rightTraingleStarsCaretHyp():
            #Objective : To display right sided triangle with stars
            and caret hypotenuse
            #Input Parameters : None
            #Return Value: None
```

```
            n=int(input('Enter number: '))
            for x in range(0,n):
                print('  '*(n-x-1)+'^ '+'* '*(x))

        rightTraingleStarsCaretHyp()
xxvii.  def rightTriangleStarsCarets():
            #Objective : To display right sided triangle with stars
            and hypotenuse of carets
            #Input Parameters : None
            #Return Value: None

            n=int(input('Enter number: '))
            for x in range(0,n):
                if x==0:
                    print('  '*(n-1)+'^')
                elif x==n-1:
                    print('^ '*n)
                else:
                    print('  '*(n-x-1)+'^ '+'* '*(x-1)+'^')

        rightTriangleStarsCarets()
xxviii. def rightTriangleHollowCaret():
            #Objective : To display hollow right sided triangle with
            carets
            #Input Parameters : None
            #Return Value: None

            n=int(input('Enter number: '))
            for x in range(0,n):
                if x==0:
                    print('  '*(n-1)+'^')
```

```
            elif x==n-1:
                print('^ '*n)
            else:
                print('  '*(n-x-1)+'^ '+  '*(x-1)+'^')

    rightTriangleHollowCaret()
```

xxix.

```
    def invertedTriangleStars():
        #Objective : To display inverted triangle with stars
        #Input Parameters : None
        #Return Value: None


        n=int(input('Enter number: '))
        for x in range(n,0,-1):
            print('* '*x)

    invertedTriangleStars()
```

xxx.

```
    def invertedStarCaretHyp():
        #Objective : To display inverted star triangle with caret
        hypotenuse
        #Input Parameters : None
        #Return Value: None

        n=int(input('Enter number: '))
        for x in range(n,0,-1):
            print('* '*(x-1)+'^')

    invertedStarCaretHyp()
```

xxxi.

```
    def invertedTriangleCaretStars():
        #Objective : To display inverted star triangle with sides
        in carets
        #Input Parameters : None
        #Return Value: None
```

```
            n=int(input('Enter number: '))
            for x in range(n,0,-1):
                if x==n:
                    print('^ '*n)
                elif x==1:
                    print('^')
                else:
                    print('^ '+'* '*(x-2)+'^')

        invertedTriangleCaretStars()
xxxii.  def invertedHollowTriangleCaret():
            #Objective : To display hollow inverted caret triangle
            #Input Parameters : None
            #Return Value: None

            n=int(input('Enter number: '))
            for x in range(n,0,-1):
                if x==n:
                    print('^ '*n)
                elif x==1:
                    print('^')
                else:
                    print('^ '+'  '*(x-2)+'^')

        invertedHollowTriangleCaret()
xxxiii. def hollowSquare():
            #Objective : To display hollow sqaure
            #Input Parameters : None
            #Return Value: None

            size = 5
            for i in range(size):
```

```
            for j in range(size):
                if i == 0 or i == size - 1 or j == 0 or j == size
                - 1:
                    print('*', end='')
                else:
                    print(' ', end='')
            print()
hollowSquare()
```

Assertion and Reasoning Based Questions

1. d

Case-based Questions

1. Do It Yourself
2. Do It Yourself

11. Modules

Assessment

A.	1.	c	2.	d	3.	c	4.	d	5. c
	6.	b	7.	c	8.	b	9.	c	10. b
B.	1.	True	2.	False	3.	True	4.	False	5. True
C.	1.	dot	2.	random	3.	largest	4.	0,1	5. right

D. 1. Default Parameters:

- Default parameters take the default values provided in the function definition when the default parameters are not specified in a function call.
- When the values of default parameters are specified in a function call, the default values provided in the function definition are ignored.
- All non-default parameters (if any) should precede the default parameters.

Keyword Arguments:

- The input arguments to a function given in an arbitrary order by explicitly associating the names of the formal parameters with their values are keyword arguments.
- Keyword arguments are often used when the default parameter list is quite long, and we wish to assign a user-specified value for only a few of them.

2. a. multiply(num1 = 6, num3 = 7)
   b. multiply(num1 = 4)
3. (a), (c), (d), (f), (g)
4. 23 ### 20
   23 ### 20
   5 ### 2
   23 ### 5
5. 120
   120
6. (10, 2, 1)
7. a. random.randint(1,500)
   b. random.randrange(5,500,5)
   c. round(math.pi,4)
   d. math.log2(100)
   e. math.log10(1000)
   f. math.log(1000)
   g. math.sin(math.pi/2)
   h. math.cos(math.pi/2)
   i. math.gcd(20,34)
   j. statistics.mean([50,54,60,61,65])
   k. math.factorial(10)
8.
```
import random
def countRandom():
    count0 = 0
    count1 = 0
    for count in range(20):
        num = random.randint(0,1)
        print(num)
        if num == 0:
            count0 = count0 + 1
        else:
            count1 = count1 + 1
    print("The count of 0s is :  ", count0)
```

```
    print("The count of 1s is :  ", count1)

countRandom()
```

9. The following programs are given in chapter 10, therefore not creating the code

(i) Half Pyramid

```
Program 10.11 in book (Do It Yourself)
```

(ii) Inverted Half Pyramid

```
def invertedHalfPyramid(nRows):
    for i in range(nRows, 0, -1):
        for j in range(0, i):
            print("* ", end=" ")
        print()
rows = int(input("Enter number of rows: "))
invertedHalfPyramid(rows)
```

(iii) Hollow Inverted Half Pyramid

```
#Hollow inverted Pyramid
def hollowInvertedHalfPyramid(nRows):
    for i in range(nRows, 0, -1):
        for j in range(0, i):
            if j == 0 or j == i - 1 or i == nRows:
                print('*', end=" ")
            else:
                print(" ", end=" ")
        print()
rows = int(input("Enter number of rows: "))
hollowInvertedHalfPyramid(rows)
```

(iv) Full Pyramid

```
Program 10.14 in book (Do It Yourself)
```

(v) Inverted Full Pyramid

```
Program 10.13 in book (Do It Yourself)
```

(vi) Hollow Full Pyramid

```
# Hollow Full Pyramid
def hollowFullPyramid(nRows):
```

```
        for i in range(nRows):
            print(' ' * (nRows - i - 1), end = ' ')
            for j in range(i + 1):
                if j == 0 or j == i or i == nRows - 1:
                    print('*', end=" ")
                else:
                    print(" ", end=" ")
            print()
rows = int(input("Enter number of rows: "))
hollowFullPyramid(rows)
```

Assertion and Reasoning Based Questions

1. a 2. a 3. b

Case-based Questions

1. Do It Yourself
2. Do It Yourself

12. Strings

Assessment

A. 1. c 2. d 3. a 4. a 5. d
6. b 7. a 8. a 9. b 10. d

B. 1. True 2. True 3. False 4. False 5. True
6. False

C. 1. 0 2. len() 3. False 4. \ 5. index 6. \n

D. 1. To access a character in a string, use the index of the required element within square brackets. Position of a character in a string is called its index. 'l', 'l', and 'o' are stored at index 0, 1, 2, 3, 4, respectively. To access a character in a string, the string object is followed by an opening square bracket, the index to be accessed, and a closing square bracket, as shown below:

```
>>> greet = 'hello'
>>> greet[0]
'h'
```

Another way to access the characters in a string is to use the negative indices -1, -2, -3,... the last character of the string being at index -1. For example,

```
>>> print(greet[-5], greet[-4], greet[-3], greet[-2], greet[-1])
hello
```

2. As string is an immutable data type, the elements of a string cannot be changed using indexes.
3. (i) `7`

   (ii) `I have 3412 fruits`

   (iii) `No output - as no print statement is given`
4. (i) `len(poem)`

   (ii) `poem.split('.')`

   (iii) `poem.split('and')`

   (iv) `poem.count('beautiful')`

   (v) poem.index('beautiful')

   (vi) `poem.rfind('beautiful')`
5. Insert a slash (\) before single quote(') as shown below:

```
msg = 'Try to be a rainbow in someone else\'s cloud.'
```

6. (i) `are o`

   (ii) `e of dog no 2`

   (iii) `no 2`

   (iv) `True`

   (v) `Beware of dog no 2`

   (vi) `False`

   (vii) `2nofew`

   (viii) `No output`

   (ix) `on god fo e`

   (x) `['Beware', 'of', 'dog', 'no', '2']`
7. (i) `-1`

   (ii) `u**a***#`
8. `find()`
   - The method find(subStr) searches for the first occurrence of the given string subStr in the string s and returns the index of the first occurrence of the string passed as an argument.
   - If the string subStr does not appear anywhere in the string s, the method find() returns -1 indicating the condition substring not found

index()

- The method index(subStr) returns the index of the first occurrence of the substring passed as an argument.
- If the string specified as the argument is not found in the given string, Python raises an exception.

9. (i) `MyString.rstrip()`

   (ii) `MyString.title()`

   (iii) `MyString.isupper()`

   (iv) `MyString.split()`

   (v) `MyString.find(String)`

   (vi) `MyString.upper()`

   (vii) Using slicing:

```
MyString = MyString[::-1]
```

10.
```
newString = ''
def replaceChar(MyString, ch):
    #Objective : To replace each occurance of character with a '*'
    #Input Parameters : MyString, ch - String Values
    #Return Value: None
    newString = ''
    for x in MyString:
        if x == ch:
            newString = newString + '*'
        else:
            newString = newString + x
    print("The New String is : ", newString)
String = input("Enter a String :  ")
character =  input("Enter a Character:  ")
replaceChar(String, character)
```

11.
```
def countWords(myString):
    #Objective : To count the number of words in a string
    #Input Parameters : MyString - String Value
    #Return Value: None
    wordList = myString.split()
```

```
        words = len(wordList)
        print(" The number words in the given string are : ", words)
    String = input("Enter a String :  ")
    countWords(String)
```

12.
```
    def countVowels(myString):
        #Objective : To count the number of vowels in a string
        #Input Parameters : MyString - String Value
        #Return Value: None
        count = 0
        for ch in myString:
            if ch in "AEIOUaeiou":
                count = count + 1
        print(" The number vowels in the given string are : ", count)
    String = input("Enter a String :  ")
    countVowels(String)
```

13.
```
    def reverseWord(myString):
        #Objective : To reverse the characters of each word in a string
        #Input Parameters : MyString - String Value
        #Return Value: None
        wordList = myString.split()
        for word in wordList:
            revWord = word[::-1]
            print(revWord, end = ' ')
    String = input("Enter a String :  ")
    reverseWord(String)
```

14.
```
    def wordString(myString):
        #Objective : To display no. of words beginning with vowel, no. of
        3-lettered words,no.of digits
        #Input Parameters : MyString - String Value
        #Return Value: None
        countVowel = 0
        count3 = 0
        countDigit = 0
```

```
    wordList = myString.split()
    for word in wordList:
        if word[0] in "AEIOUaeiou":
            countVowel += 1
        if len(word) == 3:
            count3 += 1
    for ch in myString:
        if ch in "0123456789":
            countDigit += 1
   print("The number of words beginning with a vowel are : ", countVowel)
    print("The number of 3-lettered words are  : ", count3)
    print("The number of digits are  : ", countDigit)
String = input("Enter a String :  ")
wordString(String)
```

15.
```
def changeCase(myString):
    #Objective : To change the case of each alphabet in a string
    #Input Parameters : MyString - String Value
    #Return Value: None
    newString = ''
    for ch in myString:
        if ch.isupper():
            newString += ch.lower()
        elif ch.islower():
            newString += ch.upper()
        else:
            newString += ch
    print("The new string with reversed case is :  ", newString)
String = input("Enter a String :  ")
changeCase(String)
```

16. (i)
```
def findSubstring(myString, subStr):
    #Objective : To find the first occurance of substring in a
    string
    #Input Parameters : MyString, subStr - String Values
```

```
    #Return Value: index value i - numeric value
    stringLength = len(myString)
    subLength = len(subStr)

    #If substring is longer than main string, no need to check
    if subLength > stringLength:
        return -1

    for i in range(stringLength - subLength + 1):
        match = True
        for j in range(subLength):
            if myString[i + j] != subStr[j]:
                match = False
                break
        if match:
            return i

    # Return -1 if no match is found
    return -1
String = input("Enter a String :  ")
subString = input("Enter a sub string :  ")
position = findSubstring(String, subString)
print("The substring starts at ", position + 1, " position")
```

(ii)
```
def findLastSubstring(myString, subStr):
    #Objective : To find the last occurance of substring in a
    string
    #Input Parameters : MyString, subStr - String Values
    #Return Value: index value i - numeric value
    stringLength = len(myString)
    subLength = len(subStr)
    lastPosition = -1

    #If substring is longer than main string, no need to check
```

```
    if subLength > stringLength:
        return -1

    for i in range(stringLength - subLength + 1):
        match = True
        for j in range(subLength):
            if myString[i + j] != subStr[j]:
                match = False
                break
        if match:
            lastPosition = i
    return lastPosition
String = input("Enter a String :  ")
subString = input("Enter a sub string :  ")
position = findLastSubstring(String, subString)
print("The last substring position is  ", position + 1,)
```

(iii)

```
def countSubstring(myString, subStr):
    #Objective : To find frequency of substring in a string
    #Input Parameters : MyString, subStr - String Values
    #Return Value: count - numeric value
    stringLength = len(myString)
    subLength = len(subStr)
    lastPosition = -1
    count = 0

    #If substring is longer than main string, no need to check
    if subLength > stringLength:
        return -1

    for i in range(stringLength - subLength + 1):
        match = True
        for j in range(subLength):
            if myString[i + j] != subStr[j]:
                match = False
```

```
                    break
            if match:
                count += 1
        return count
    String = input("Enter a String :  ")
    subString = input("Enter a sub string :  ")
    cnt = countSubstring(String, subString)
    print("The count of substring in the string is   ", cnt,)
```

(iv)
```
    def countConsonant(myString):
        #Objective : To find the number of consonants in a string
        #Input Parameters : MyString - String Value
        #Return Value: count - numeric value
        count = 0
        for ch in myString:
            if ch not in "AEIOUaeiou ":
                count += 1
        return count
    String = input("Enter a String :  ")
    cnt = countConsonant(String)
    print("The number of consonants in the string is   ", cnt)
```

(v)
```
    def countAlpha(myString):
        #Objective : To count the number of uppercase and lowercase
        characters
        #Input Parameters : MyString - String Value
        #Return Value: None
        countU = 0
        countL = 0
        for ch in myString:
            if ch >= 'A' and ch <= 'Z':
                countU += 1
            elif ch >= 'a' and ch <= 'z':
                countL += 1
        print("The number of uppercase letters is  ", countU)
```

```
        print("The number of lowercase letters is  ", countL)
    String = input("Enter a String :  ")
    countAlpha(String)
```

(vi)
```
    def frequentChar(myString):
        #Objective : To find the most frequently occuring character
        #Input Parameters : MyString - String Value
        #Return Value: None

        maxCount = 0
        mostFrequent = None
        for i in range(len(myString)):
            ch = myString[i]
            count = 0

            for j in range(len(myString)):
                if myString[j] == ch:
                    count += 1

            # Update most frequent character if needed
            if count > maxCount:
                maxCount = count
                mostFrequent = ch

        print("The most frequent character is  ", mostFrequent)
    String = input("Enter a String :  ")
    frequentChar(String)
```

(vii) Same as answer 13 (Do It Yourself)

Assertion and Reasoning Based Questions

1. a 2. b

Case-based Questions

1. Do It Yourself
2. Do It Yourself

13. Lists and Tuples

Assessment

A. 1. d 2. b 3. c 4. c 5. c
6. b 7. a 8. c 9. b 10. d

B. 1. True 2. False 3. True 4. False 5. True

C. 1. squarebrackets [] 2. list() 3. three 4. sorted()
5. +

D. 1.

```
def maxElement(myList):
    #Objective : To find the largest element from the list
    #Input Parameters : myList - List Value
    #Return Value: maxNum - numeric value

    maxNum = myList[0]
    for x in myList:
        if x > maxNum:
            maxNum = x
    return maxNum
def acceptList():
    #Objective : To accept data and form a list
    #Input Parameters : None
    #Return Value: None

    L = []
    for x in range(6):
        num = int(input("Enter a number : "))
        L.append(num)
    largestNumber = maxElement(L)
    print("The largest number is  ", largestNumber)
acceptList()
```

2.

```
def minElement(myTuple):
    #Objective : To find the largest element from the tuple
```

```
    #Input Parameters : myTuple - Tuple Value
    #Return Value: minNum - numeric value

    minNum = myTuple[0]
    for x in myTuple:
        if x < minNum:
            minNum = x
    return minNum
def acceptTuple():
    #Objective : To accept data and form a tuple
    #Input Parameters : None
    #Return Value: None

    L = []
    for x in range(6):
        num = int(input("Enter a number : "))
        L.append(num)
    T = tuple(L)
    smallestNumber = minElement(T)
    print("The smallest number is  ", smallestNumber)
acceptTuple()
```

3.
```
def displayDate():
    #Objective : To display date in the given format
    #Input Parameters : None
    #Return Value: T - tuple value
    T = ()
    s = input("Enter a date in ddmmyyyy format : ")
    if len(s) != 8:
        print("Date not entered correctly")

    else:

        T = T + (s[:2],)
```

```
            T = T + (s[2:4],)
            T = T + (s[4:],)
        return T
    date = displayDate()
    if len(date) != 0:
        print("Day :  ", date[0])
        print("Month :  ", date[1])
        print("Year :  ", date[2])
```

4.
```
    def fiboTuple(n):
        #Objective : To form a tuple comprising the first n elements of
        the Fibonacci sequence:
        #Input Parameters : n - numeric value
        #Return Value: fibTuple - tuple value


        a = 0
        b = 1
        fibTuple = (0, 1)
        for x in range(n - 2):
            z = a + b
            fibTuple = fibTuple + (z,)
            a = b
            b = z
        return fibTuple
    numElements = int(input("Enter the number of elements of a Fibonacci
    series : "))
    fibSeries = fiboTuple(numElements)
    for num in fibSeries:
        print(num, end = '  ')
```

5. 8

6.
```
    def swapElements(myList):
        #Objective : To swap adjacent elements of a list
        #Input Parameters : myList - list value
        #Return Value: None
```

```
    print(" The current list is : \t ", myList)
    for x in range(0,len(myList)- 1,2):
        myList[x], myList[x+1] = myList[x+1], myList[x]
    print(" The new list is : \t ", myList)

def acceptList():
    #Objective : To accept data and form a list
    #Input Parameters : None
    #Return Value: None

    L = []
    for x in range(6):
        num = int(input("Enter a number : "))
        L.append(num)
    swapElements(L)
acceptList()
```

7.
```
def swapElements1(myList):
    #Objective : To swap adjacent elements of a list
    #Input Parameters : myList - list value
    #Return Value: None
    print(" The current list is : \t ", myList)
    if len(myList) % 2 == 0:
        for x in range(0,len(myList)- 1,2):
            myList[x], myList[x+1] = myList[x+1], myList[x]
    else:
        for x in range(0,len(myList)- 2,2):
            myList[x], myList[x+1] = myList[x+1], myList[x]

    print(" The new list is : \t ", myList)
def acceptList():
    #Objective : To accept data and form a list
    #Input Parameters : None
    #Return Value: None
```

```
    L = []
    numList = int(input("Enter the number of elements for the list"))
    for x in range(numList):
        num = int(input("Enter a number : "))
        L.append(num)
    swapElements1(L)
acceptList()
```

8.
```
def tupleElements(myTuple):
    #Objective : To construct a tuple with
    #   The elements at odd indexes are same as those at odd indexes
    in the tuple tpl.
    #The elements at even indexes are computed  according to the formula:
    n**2, where n is the index
    #Input Parameters : myTuple - tuple value
    #Return Value: None
    print("The current tuple is : ", myTuple)
    sqrTuple = ()
    for x in range(len(myTuple)):
        if x % 2 != 0:
            sqrTuple = sqrTuple + (myTuple[x],)
        else:
            sqrTuple = sqrTuple +  (x ** 2,)
    print("The new tuple is : ", sqrTuple)
def acceptTuple():
    #Objective : To accept data and form a tuple
    #Input Parameters : None
    #Return Value: None

    L = []
    numRange = int(input("Enter the number of elements for the tuple
    :  "))
    for x in range(numRange):
        num = int(input("Enter a number : "))
```

```
        L.append(num)
    T = tuple(L)
    tupleElements(T)
acceptTuple()
```

9.
```
def tupleToList(myTuple):
    #Objective : To construct a list with
    #   The elements at odd indexes are same as those at odd indexes
    in the tuple tpl.
    #The elements at even indexes are computed  according to the formula:
    n**2, where n is the index
    #Input Parameters : myTuple - tuple value
    #Return Value: None
    print("The current tuple is : ", myTuple)
    sqrList = []
    for x in range(len(myTuple)):
        if x % 2 != 0:
            sqrList.append(myTuple[x])
        else:
            sqrList.append(x ** 2)
    print("The new list is : ", sqrList)
def acceptTuple():
    #Objective : To accept data and form a tuple
    #Input Parameters : None
    #Return Value: None

    L = []
    numRange = int(input("Enter the number of elements for the tuple
    :  "))
    for x in range(numRange):
        num = int(input("Enter a number : "))
        L.append(num)
    T = tuple(L)
    tupleToList(T)
acceptTuple()
```

10.
```
import math
def cubeList(myList):
    #Objective : To replace the elements at indices which are multiples
    of 3,
    # by the cube of the number at that index
    #Input Parameters : myList - list value
    #Return Value: None
    print("The current list is : ", myList)
    cList = []
    for x in range(len(myList)):
        if x % 3 == 0:
            value = int(math.pow(myList[x],3))
            cList.append(value)
        else:
            cList.append(myList[x])
    print("The new list is : ", cList)
def acceptList():
    #Objective : To accept data and form a list
    #Input Parameters : None
    #Return Value: None

    L = []
    numList = int(input("Enter the number of elements for the list"))
    for x in range(numList):
        num = int(input("Enter a number : "))
        L.append(num)
    cubeList(L)
acceptList()
```

11.
```
def linearSearch(myList, key):
    #Objective : To search for an element in a list
    #Input Parameters : myList - list value, key - numeric value
    #Return Value: index - integer value
    print("The current list is : ", myList)
```

```
        for x in range(len(myList)):
            if myList[x] == key:
                return x
        return None
    def acceptList():
        #Objective : To accept data and form a list
        #Input Parameters : None
        #Return Value: None

        L = []
        numList = int(input("Enter the number of elements for the list :
        "))
        for x in range(numList):
            num = int(input("Enter a number : "))
            L.append(num)
        print("The list your entered is : ", L)
        number = int(input("Enter the element to be searched : "))
        found = linearSearch(L, number)
        if found != None:
            print(" The element is at index : ", found)
        else:
            print("NOT FOUND")
    acceptList()
```

12.
```
    def marksUpdate(RollNoMarksList):
        #Objective : To increase the marks by 5 in a nested list
        #Input Parameters : RollNoMarksList - list value
        #Return Value:   RollNoMarksList- list value
        print("The current data is : ")
        for record in RollNoMarksList:
            print(record)
            record[1] += 5
        return RollNoMarksList
```

```
def acceptData():
    #Objective : To accept roll no and marks and store it in the form
of a nested list
    #Input Parameters : None
    #Return Value: None

    L = []
    studentData = []
    numList = int(input("Enter the number of records that you want to
enter : "))
    for x in range(numList):
        rollno = int(input("Enter Roll No : "))
        marks = int(input("Enter Marks :  "))
        L = [rollno,marks]
        studentData.append(L)
    print("The records you entered are : ", studentData)
    updatedRecords = marksUpdate(studentData)
    print("The updated records are : ")
    for rec in updatedRecords:
        print(rec)
acceptData()
```

13.
```
def minorModeration(NameMarksList):
    #Objective : To award pass parks but by not more than three marks
    #Input Parameters : NameMarksList - list value
    #Return Value:   NameMarksList- list value
    passMarks = int(input("Enter passMarks :  "))
    print("The current data is : ")
    for record in NameMarksList:
        print(record)
        diff = passMarks - record[1]
        if diff <= 3:
            record[1] += diff
    return NameMarksList
```

```
def acceptData():
    #Objective : To accept roll no and marks and store it in the form
    of a nested list
    #Input Parameters : None
    #Return Value: None

    L = []
    studentData = []
    numList = int(input("Enter the number of records that you want to
    enter : "))
    for x in range(numList):
        name = input("Enter Name : ")
        marks = int(input("Enter Marks :  "))
        L = [name,marks]
        studentData.append(L)
    print("The records you entered are : ", studentData)
    updatedRecords = minorModeration(studentData)
    print("The updated records are : ")
    for rec in updatedRecords:
        print(rec)
acceptData()
```

14.
```
def moderation(NameMarksList):
    #Objective : To award pass parks but by not more than three marks
    #Input Parameters : NameMarksList - list value
    #Return Value:   NameMarksList- list value
    print("The current data is : ")
    for record in NameMarksList:
        print(record)
        if record[1] < 40:
            record[1] += 6
        elif record[1] > 40 and record[1] <= 50:
            record[1] += 5
        elif record[1] > 50 and record[1] <= 60:
```

```
            record[1] += 4
        elif record[1] > 60 and record[1] <= 70:
            record[1] += 3
        elif record[1] > 70 and record[1] <= 80:
            record[1] += 2
        elif record[1] > 80 and record[1] <= 90:
            record[1] += 1
        elif record[1] > 90:
            record[1] += 0

    return NameMarksList
def acceptData():
    #Objective : To accept roll no and marks and store it in the form
    of a nested list
    #Input Parameters : None
    #Return Value: None

    L = []
    studentData = []
    numList = int(input("Enter the number of records that you want to
    enter : "))
    for x in range(numList):
        name = input("Enter Name : ")
        marks = int(input("Enter Marks :  "))
        L = [name,marks]
        studentData.append(L)
    print("The records you entered are : ", studentData)
    updatedRecords = moderation(studentData)
    print("The updated records are : ")
    for rec in updatedRecords:
        print(rec)
acceptData()
```

Assertion and Reasoning Based Questions

1. a 2. c 3. b

Case-based Questions

1. Do It Yourself
2. Do It Yourself

14. Dictionaries

Assessment

A. 1. c 2. d 3. a 4. a 5. b 6. d
7. b 8. b 9. d 10. a 11. a

B. 1. True 2. True 3. False 4. True 5. True

C. 1. curly {} 2. key 3. del 4. dict() 5. copy()

D. 1. A dictionary is an unordered set of key:value pairs. A dictionary (an object of type dict) is defined in Python by enclosing the comma-separated key: value pairs in braces. An empty pair of braces denotes an empty dictionary. For example,

```
ticketPrice = {'Chandigarh':450, 'Nagpur':760, 'Pune':380}
```

2. No, because dictionary is not an ordered data type.
3. Objects of the immutable types, strings, numbers, and tuples are used as keys.
4. • using dict(), for example, D = dict()
   • using a pair of curly brackets. For example, D = {}
5. A dictionary is an unordered set of key:value pairs. A dictionary (an object of type dict) is defined in Python by enclosing the comma-separated key: value pairs in braces. The values are corresponding to the keys. The values are referred through the keys instead of indexes.
6.
```
rollnoMarks = {1:60,2:25,3:90,4:32,5:12,6:78}
for k in rollnoMarks:
    if rollnoMarks[k] >= 40:
        print(k)
```
7.
```
def charCountDict(string):
    #Objective : To count the number of occurances of each character
    in a string
    # and form a dictionary
```

```
    #Input Parameters : string - string value
    #Return Value: None

    charCount = {}
    for ch in string:
        if ch in charCount:
            charCount[ch] += 1
        else:
            charCount[ch] = 1
    print("The dictionary with each character count in the ", string,
    "  is : ", charCount)

def acceptStrings():
    #Objective : To accept strings and store it in the form of a list
    #Input Parameters : None
    #Return Value: None

    stringList = []

    num = int(input("Enter the number of strings that you want to enter
    : "))
    for x in range(num):
        s = input("Enter a string : ")
        stringList.append(s)
    print("The strings you entered are : ", stringList)
    print("The character wise count of each string is : ")
    for rec in stringList:
        charCountDict(rec)
acceptStrings()
```

8.
```
def dictStudents():
    #Objective : To store the names and marks and store it in the form
    of a dictionary
    # and display name of the student with maximum marks
    #Input Parameters : None
```

```
    #Return Value: None
    studentData = {}
    num = int(input("Enter the number of records that you want to enter
    : "))
    for x in range(num):
        print("Record : ", x + 1)
        name = input("Enter name : ")
        marks = int(input("Enter marks : "))
        studentData[name] = marks
    print("The records entered are : \n", studentData)
    maxMarks = 0
    for x in studentData:
        if studentData[x] > maxMarks:
            maxMarks = studentData[x]
            maxName = x

    print("The student with maximum marks is ", maxName)

dictStudents()
```

9.
```
months = {}
def dictMonths():
    #Objective : To create a dictionary of month names and the number
    of days in a month.
    #Input Parameters : None
    #Return Value: None

    for x in range(12):
        print("Record : ", x + 1)
        mName = input("Enter name of month : ")
        mDays = int(input("Enter number of days : "))
        months[mName] = mDays
def days31():
    #Objective : To display the names of months with 31 days
```

```
    #Input Parameters : None
    #Return Value: None
    print("The months with 31 days are : ")
    for x in months:
        if months[x] == 31:
            print(x)
def days():
    #Objective : To accept month name from a user and display the
    corresponding number of days.
    #Input Parameters : None
    #Return Value: None
    nameMonth = input("Enter the name of the month : ")
    for x in months:
        if x == nameMonth:
            print("The number of days in ", x, " is ", months[x])
            break
def sortedMonths():
    #Objective : To display the names of months in alphabetical order.
    #Input Parameters : None
    #Return Value: None

    L = sorted(months)
    print("The names of months in alphabetical order are : ")
    for mnth in L:
        print(mnth)


dictMonths()
days31()
days()
sortedMonths()
```

10.
```
employeeData = {}
def acceptData():
    #Objective : To store the adhar card numbers, names and saralies
```

```
    of employees in a dictionary
    #Input Parameters : None
    #Return Value: None

    num = int(input("Enter the number of records that you want to enter
    : "))
    for x in range(num):
        print("Record : ", x + 1)
        aadhar = int(input("Enter Aadhaar card number : "))
        name = input("Enter name : ")
        salary = int(input("Enter salary : "))
        employeeData[aadhar] = [name, salary]
    print("The records entered are : \n", employeeData)
    mainMenu()

def dispData():
    #Objective : To accept aadhar card number and display details of
    employees
    #Input Parameters : None
    #Return Value: None
    aadhar = int(input("Enter Aadhaar card number : "))
    for record in employeeData:
        if record == aadhar:
            print("The details of employee are : ")
            print("Aadhar Card No : ", record)
            print("Name           : ", employeeData[record][0])
            print("Salary           : ", employeeData[record][1])
                    break
    else:
        print("No record found !!!!!!\n\n")

    mainMenu()
```

```
def mainMenu():
    #Objective : To display main menu and call fucntions as per the
    choice
    #Input Parameters : None
    #Return Value: None
    print("1. Accept Data \n 2. Display Data\n 3. Exit")
    ch = int(input("enter your choice : "))
    if ch == 1 :
        acceptData()
    elif ch == 2:
        dispData()
    else:
        return
mainMenu()
```

11.
```
employeeData = {}
def acceptData():
    #Objective : To store the employee id, name, department and salary
    of employees in a dictionary
    #Input Parameters : None
    #Return Value: None

    num = int(input("Enter the number of records that you want to enter
    : "))
    for x in range(num):
        print("Record : ", x + 1)
        empId = int(input("Enter Employee Id : "))
        name = input("Enter name : ")
        dept = input("Enter department name : ")
        salary = int(input("Enter salary : "))
        employeeData[empId] = [name, dept, salary]
    print("The records entered are : \n", employeeData)


def highSalary():
    #Objective : To display id and name of employees having salary
```

```
    greater than 50000
    #Input Parameters : None
    #Return Value: None
    print("The employees with salary greater than 50000 are : ")
    for record in employeeData:
        if employeeData[record][2] > 50000:
            print("Employee Id : ", record)
            print("Name        : ", employeeData[record][0])


def dispData():
    #Objective : To accept employee id and display teh corresponding
    record
    #Input Parameters : None
    #Return Value: None
    eId = int(input("Enter Employee Id : "))
    for record in employeeData:
        if record == eId:
            print("The details of employee are : ")
            print("Employee Id  : ", record)
            print("Name           : ", employeeData[record][0])
            print("Department          : ", employeeData[record][1])
            print("Salary          : ", employeeData[record][2])
            break
    else:
        print("No record found !!!!!!\n\n")


def sortedId():
    #Objective : To display the ids of employees in sorted order.
    #Input Parameters : None
    #Return Value: None

    L = sorted(employeeData)
    print("The id of employees in sorted order are : ")
```

```
        for id in L:
            print(id)

acceptData()
highSalary()
dispData()
sortedId()
```

Assertion and Reasoning Based Questions

1. a 2. b

Case-based Questions

1. Do It Yourself
2. Do It Yourself

15. Society, Law and Ethics

Assessment

A. 1. c 2. a 3. c 4. b 5. d
6. b 7. b

B. 1. True 2. False 3. False 4. False 5. True
6. True 7. True 8. True

C. 1. trojan horse 2. denial of service / DoS 3. copyright 4. 20
5. Digital Rights Management 6. licence 7. Cyber Law 8. e-waste
9. Braille

D. 1. a. A Denial of Service, or DOS attack renders a machine or network resource unavailable either temporarily or permanently to its intended users. A DOS attack is typically carried out by sending a lot of false requests to the targeted machine or resource, which eventually cause the system to become too busy to work.

b. When someone passes off the words, ideas, or expressions of another author as their own, it is an act of plagiarism.

c. Privacy law establishes the rules that regulate the collection, storage, and disclosure of a person's or organisation's financial, medical, and other personal information to third parties.

d. The use or creation of copyright-protected works without the owner's consent is copyright infringement.

e. Electronic products that are no longer needed, are broken, or have reached the end of their useful lives are called E-waste.

2. Attacks that steal data being transmitted over a communication channel are known as eavesdropping.

3. a. File Viruses:

- Viruses that attach themselves to executable files by overwriting a part of their code or appending their code to the files.
- For example, the Romeo and Juliet virus.

Macro Virus:

- Macro viruses embed themselves into the documents.
- These are executable files that may be received as email attachments. When a recipient opens an attachment, the viruses gets activated and starts affecting the system programs (deleting, creating, or overwriting other files).
- For example, the Melissa Virus got spread through a Microsoft Word document sent as an attachment.

b. Adware:

- An adware is a software that displays advertisements to users of the Internet and mobile applications.
- Adware may also record user activities and pass them on to the relevant businesses for money.
- Adware may also act as spyware by collecting private and sensitive information.

Spyware:

- The spyware secretly collects information from a system.
- Some common signs of a system under the spyware attack are: getting constant error messages, random icons being displayed on the desktop,

c. Copyright:

- When an author or an organisation sets a work in a physical form of expression, they have a copyright on it that prevents others from claiming its ownership.
- Some examples of such intellectual work are drawings, pictures, illustrations, musical compositions, sound recordings, computer programs,

Patent:

- A patent is a property right that an autonomous body (or a sovereign state) grants to the inventor for a specific amount of time.
- It provides a detailed description of an invention in the form of a document. It makes sure that the patented invention can only be produced, imported, used, and sold with the permission of the patent owner.
- The term of a patent is usually 20 years, starting from the date of the patent application.

d. Proprietary License:

- The users of software under proprietary licence will be able to use the software, but they do not have access to the source code and are not authorised to modify it.
- Source code is not open to the users. Some proprietary software may be free to use, but you cannot access the source code.
- Examples of the priced proprietary software include Windows, Mac, and Microsoft Office.

Free and Open Source License:

- It is the software that comes with publicly available source code.
- Open source licence allows the user to modify the software and distribute the modified version.
- It is typically developed in a collaborative manner by a community of developers.
- Examples are Python, Open Office, etc.

4. a. Romeo and juliet Virus
   b. Code Red Worm
   c. Grammarly
5. **IPR:** Intellectual Property Rights

   DRM: Digital Rights Management

   GPL: General Public License
6. When you visit a website, it may store small pieces of text, called cookies so that it can keep track of the users' preferences, and provide the users a more satisfying experience. A cache is employed to store the website content and page resource in the browser for long run purposes. It helps to decrease the loading time of a web page.
7. - It is accessible in a user-friendly and adaptable format.
   - The data must be freely available for download via the internet.
   - The data must be re-usable and redistributable.
8. A software licence states the rights of the developer. It includes how the software should be used. It also includes some terms and conditions about how the software should be used, as well as limitations on liability, disclaimers, and any warranties that apply.

9. Proprietary licence is also known as closed source licence because the source code is not open to the users.

1.0. Creative Commons licences give content creators a standard way to let other people use their work. It is a standard form of a licence agreement that lets someone use a work without having to talk to the creator or negotiate terms. For example, filmmakers can use a Creative Commons CC BY licence for their videos on YouTube. Thus, they retain their copyright on the content that can be freely used by their users.

11. Creative Commons License, General Piblic License

7. • It is accessible in a user-friendly and adaptable format.

12. • **Landfills:** The e-waste is buried in deep dirt trenches
    - **Incineration:** The incineration process involves burning e-waste at high temperatures in incinerators.
    - **Acid Bath:** In this method, the e-waste is soaked in solutions of sulphuric, hydrochloric, and nitric acids to get the metal out. The salvaged metal is then used to make other things.
    - **Mechanical recycling:** The most effective and environmentally responsible method is to recycle. In this method, used circuit boards, ICs, motherboards, and other parts are taken from e-waste and recycled by dry physical separation.

13. Social Impact
    - From education to entertainment to health to how we celebrate holidays and how we talk to our friends, family, and coworkers.
    - IT has opened up new entertainment options
    - Excessive online gaming and other forms of entertainment are bad for our health because they make us less active.
    - IT has also introduced new avenues for criminals in the form of cybercrimes.

    Cultural Impact
    - More online meetings than physical meetings. Reduced human interaction has led to loneliness, which may lead to minor to severe depression.
    - Headphones are preferred to open ears for listening to music.
    - People would rather shop online than at a store near them, which hurts small businesses.

14. Yes, girls in our society are not encouraged to pursue technical courses. The reasons are:
    - There is an unsubstantiated belief in some sections of the society that the boys are better at technical things while girls are better at humanities, arts, and so on. Further, the girls should pursue a career like teaching that requires fewer hours at work so that they can focus on family.
    - During the early years, children often play games on computers and smartphones. Since many of the games are geared toward boys, they introduce gender bias at a young age.

- Girls see fewer women as IT leaders who could be their role models. It impacts their decision about the choice of discipline.

15.
- Lack of suitable instructional materials designed to meet the needs of the students with disabilities.
- Students who are deaf or hard of hearing may need more visual input than verbal. The schools do not typically provide these facilities for reasons of financial constraints or a lack of appreciation for the specialised needs.
- The educational institutes need to train the teachers to handle the students with special needs.

Assertion and Reasoning Based Questions

1. a 2. c 3. b

Case-based Questions

1. Do It Yourself
2. Do It Yourself