Unit -5

- Current Challenges in IoT:
    - Large Scale of Cooperation:
        - Coordination among millions of distributed devices on the Internet.
    - Global Heterogeneity:
        - Variability in types of IoT devices and their respective subnets.
    - Unknown IoT Device Configuration:
        - Different configuration modes for devices that may come from unknown or varied owners.
    - Semantic Conflicts:
        - Different processing logics applied to similar networked devices or applications, leading to misunderstandings.
- Definition of Interoperability:
    - Interoperability is a characteristic of a product or system whose interfaces are completely understood, enabling it to work with other products or systems (current or future) without restrictions.
        - Key Aspects:
            - Ability to communicate meaningfully.
            - Capability to exchange data or services.
- Importance of Interoperability in IoT:
    - Essential for fulfilling IoT objectives:
        - Physical objects can interact and share information with one another.
        - Devices can communicate anytime and anywhere.
        - Supports Machine-to-Machine (M2M), Device-to-Device (D2D), and Device-to-Machine (D2M) communication.
        - Facilitates seamless integration of devices into the IoT network.
- Reasons for Interoperability Requirement:
    - Heterogeneity:
        - Various communication protocols (e.g., ZigBee (IEEE 802.15.4), Bluetooth (IEEE 802.15.1), GPRS, 6LoWPAN, Wi-Fi (IEEE 802.11)).
        - Different wired protocols (e.g., Ethernet (IEEE 802.3)).
        - Diverse programming languages (e.g., JavaScript, Java, C, C++, Visual Basic, PHP, Python).
        - Varied hardware platforms (e.g., Crossbow, NI).
        - Different operating systems (e.g., TinyOS, Windows, Mac, Unix).
        - Multiple database types (e.g., DB2, MySQL, Oracle).
        - Different data representations and control models.
- Types of Interoperability:

        - User Interoperability:
            - Issues arising between a user and a device.
        - Device Interoperability:
            - Problems encountered between two different devices.
- Example of Device and User Interoperability:
    - Scenario: Two devices (A and B) providing security services in different locations (Delhi, India, and Tokyo, Japan) with users speaking different languages (Hindi, Japanese, English).
    - Problems:
        - Lack of user knowledge about devices A and B.
        - Incompatibility in syntactic and semantic understanding between devices and user.
- Key Requirements for Interoperability:
    - Device Identification and Categorization:
        - Methods for generating unique addresses (e.g., Electronic Product Codes (EPC), Universal Product Code (UPC), Uniform Resource Identifier (URI), Internet Protocol (IP) addresses).
    - Syntactic Interoperability:
        - Ensures that message formats are understandable between devices and users.
    - Semantic Interoperability:
        - Ensures that the meaning of messages is comprehensible to both devices and users.
- Solutions for Device Interoperability:
    - Universal Middleware Bridge (UMB):
        - Addresses interoperability issues caused by the heterogeneity of various home network middleware.
        - Creates virtual maps among physical devices across different middleware networks (e.g., HAVI, Jini, LonWorks, UPnP).

Arduino Programming

- Arduino Overview:
    - An open-source electronic programmable board (microcontroller) and integrated development environment (IDE).
    - Capable of accepting both analog and digital signals as input and providing the desired output.
- Types of Arduino Boards:

        - Based on different microcontrollers:
            - ATmega328, ATmega32u4, ATmega2560, AT91SAM3X8E.
- Arduino IDE:
    - Software used for programming Arduino boards.
    - Based on variations of C and C++ programming languages.
    - Can be downloaded from the official Arduino website.
- Setting Up Arduino:
    - Connect the board to a PC via USB.
    - Launch the Arduino IDE.
    - Set the board type and port via the Tools menu.
- Sketch Structure:
    - A sketch consists of two main parts:
        - setup(): Initializes variables and pin modes (analog or digital).
        - loop(): Contains the main code that runs repeatedly.
- Supported Data Types:
    - Void, Long, Int, Char, Boolean

1. The main goal of UMB is to achieve interoperability among heterogeneous devices in home networks.
2. UMB addresses the challenge of global heterogeneity, which refers to the diversity of IoT devices and their subnets.
3. One of the current challenges in IoT is the unknown configuration of device configurations, which can vary widely among different manufacturers.
4. Semantic conflicts occur when different processing logics are applied to the same IoT networked devices.
5. Interoperability allows devices to communicate meaningfully and exchange data or services.
6. UMB creates virtual maps among the physical devices of all middleware home networks.
7. The architecture of UMB consists of two main components: UMB Core (UMB-C) and UMB Adaptor.
8. The UMB Adaptor (UMB-A) converts physical devices into virtually abstracted ones.
9. The Universal Device Template (UDT) includes a Global Device ID, Global Function ID, Global Action ID, and Global Parameters.
10. UMB facilitates communication among devices that use different communication protocols.

11. UMB aims to solve the seamless interoperability problems caused by the heterogeneity of several kinds of home network middleware.
12. The Core of UMB is responsible for routing the universal metadata message to the destination.
13. The Adaptor of UMB allows devices to be discovered and controlled remotely.
14. UMB helps in device identification and categorization for discovery.
15. Syntactic interoperability ensures that the message format from a device to a user is understandable for the user's computer.
16. Semantic interoperability allows the device to understand the meaning of user instructions.
17. The Global Device ID is a unique identifier used for device identification in UMB.
18. The UNSPSC standard is used for efficient classification of products and services in UMB.
19. UMB provides a bridge for devices using protocols such as ZigBee, Bluetooth, and Wi-Fi.
20. The concept of cooperation refers to the cooperation and coordination of millions of distributed devices.
21. UMB is designed to create compatibility among middleware home networks like HAVI, Jini, and LonWorks.
22. The Middleware Routing table is used by UMB Core to route messages.
23. UMB allows for seamless integration with IoT networks.
24. The UMB Adaptor translates local middleware's messages into global messages.
25. UMB addresses the challenge of global, which refers to the need for devices to communicate anytime from anywhere.
26. The interoperability of UMB is crucial for ensuring that devices can interact despite differences in technology.
27. UMB supports both open and proprietary protocols for device communication.
28. The ability of devices to communicate with each other is known as device-to-device communication.
29. UMB aims to reduce semantic conflicts between devices from different manufacturers.
30. The process of creating virtual maps among physical devices is called virtual mapping.
31. UMB enhances the user experience by allowing users to control multiple devices through a single interface.
32. The syntactic representation of devices is a key challenge that UMB addresses.
33. UMB helps in overcoming the incompatibility of devices that are not natively compatible.
34. The flexibility of UMB allows for dynamic interaction between devices.

35. UMB's ability to handle new devices being added to a network is essential for scalability.
36. The interoperability of UMB is crucial for ensuring that devices can communicate effectively.
37. UMB can integrate with existing home automation systems.
38. The diversity of devices is a key challenge that UMB addresses.
39. UMB helps in reducing the complexity of communication among various devices.
40. The UMB Adaptor enables devices to communicate using a standardized interface.
41. UMB enhances interoperability by allowing for interoperability across different networks.
42. The control in UMB ensures that devices can be controlled remotely.
43. UMB's ability to handle different communication protocols is essential for its functionality.
44. The routing of UMB is crucial for ensuring that devices can interact with each other seamlessly.
45. UMB's architecture helps in the abstraction of device functionalities.
46. The UMB Core utilizes a Middleware Routing Table for routing messages.
47. UMB supports the integration of devices across different middleware environments.
48. The heterogeneity of devices is a key challenge that UMB addresses.
49. UMB helps in reducing the complexity of **communication ** among various devices.
50. The UMB Adaptor enables devices to communicate using a standardized interface.
51. UMB enhances interoperability by allowing for interoperability across different networks.
52. The control in UMB ensures that devices can be controlled remotely.
53. UMB's ability to handle different communication protocols is essential for its functionality.
54. The routing of UMB is crucial for ensuring that devices can interact with each other seamlessly.
55. UMB's architecture helps in the abstraction of device functionalities.
56. The UMB Core utilizes a Middleware Routing Table for routing messages.
57. UMB supports the integration of devices across different middleware environments.
58. The heterogeneity of devices is a key challenge that UMB addresses.
59. UMB helps in reducing the complexity of communication among various devices.
60. The UMB Adaptor enables devices to communicate using a standardized interface.

61. UMB enhances interoperability by allowing for interoperability across different networks.
62. The control in UMB ensures that devices can be controlled remotely.
63. UMB's ability to handle different communication protocols is essential for its functionality.
64. The routing of UMB is crucial for ensuring that devices can interact with each other seamlessly.
65. UMB's architecture helps in the abstraction of device functionalities.
66. The UMB Core utilizes a Middleware Routing Table for routing messages.
67. UMB supports the integration of devices across different middleware environments.
68. The heterogeneity of devices is a key challenge that UMB addresses.
69. UMB helps in reducing the complexity of communication among various devices.
70. The UMB Adaptor enables devices to communicate using a standardized interface.
71. UMB enhances interoperability by allowing for interoperability across different networks.
72. The control in UMB ensures that devices can be controlled remotely.
73. UMB's ability to handle different communication protocols is essential for its functionality.
74. The routing of UMB is crucial for ensuring that devices can interact with each other seamlessly.
75. UMB's architecture helps in the abstraction of device functionalities.
76. The UMB Core utilizes a Middleware Routing Table for routing messages.
77. UMB supports the integration of devices across different middleware environments.
78. The heterogeneity of devices is a key challenge that UMB addresses.
79. UMB helps in reducing the complexity of communication among various devices.
80. The UMB Adaptor enables devices to communicate using a standardized interface.
81. UMB enhances interoperability by allowing for interoperability across different networks.
82. The control in UMB ensures that devices can be controlled remotely.
83. UMB's ability to handle different communication protocols is essential for its functionality.
84. The routing of UMB is crucial for ensuring that devices can interact with each other seamlessly.
85. UMB's architecture helps in the abstraction of device functionalities.
86. The UMB Core utilizes a Middleware Routing Table for routing messages.

87. UMB supports the integration of devices across different middleware environments.
88. The heterogeneity of devices is a key challenge that UMB addresses.
89. UMB helps in reducing the complexity of communication among various devices.
90. The UMB Adaptor enables devices to communicate using a standardized interface.
91. UMB enhances interoperability by allowing for interoperability across different networks.
92. The control in UMB ensures that devices can be controlled remotely.
93. UMB's ability to handle different communication protocols is essential for its functionality.
94. The routing of UMB is crucial for ensuring that devices can interact with each other seamlessly.
95. UMB's architecture helps in the abstraction of device functionalities.
96. The UMB Core utilizes a Middleware Routing Table for routing messages.
97. UMB supports the integration of devices across different middleware environments.
98. The heterogeneity of devices is a key challenge that UMB addresses.
99. UMB helps in reducing the complexity of communication among various devices.
100. The UMB Adaptor enables devices to communicate using a standardized interface.

https://www.blackbox.ai/chat/Ack6WpC

UNSPSC (United Nations Standard Products and Services Code)

Wek-6

1. Python is a _____ language known for its readability and ease of use. (versatile)
2. The primary data types in Python include _____, Strings, Lists, Tuples, and Dictionaries. (Numbers)
3. To print a statement in Python, you use the _____ function. (print)
4. A _____ is a segment of code that performs a specific task and can be reused. (function)
5. The syntax for defining a function in Python is _____ . (def function_name(args):)
6. In Python, the _____ statement is used to handle exceptions. (try-except)

7. The command to install the GPIO library on Raspberry Pi is _____ . (sudo apt-get install python-rpi.gpio)
8. The primary purpose of a sensor is to convert _____ quantities into electrical signals. (physical)
9. The _____ is a mechanical device that converts energy into motion. (actuator)
10. Raspberry Pi is a _____-cost, single-board computer ideal for IoT projects. (low)
11. The command to read from a file in Python is _____ . (file.read())
12. The mode 'r' in the open function stands for _____ mode. (read)
13. To close a file in Python, you use the _____ method. (close)
14. The _____ library is used for image processing in Python. (Pillow)
15. The function to handle CSV files in Python is _____ . (csv.reader and csv.writer)
16. In socket programming, the socket family can be either AF_INET or _____ . (AF_UNIX)
17. The command to create a TCP/IP socket in Python is _____ . (socket.socket())
18. The _____ command is used to enable SSH on Raspberry Pi. (sudo raspi-config)
19. The GPIO pin numbering mode can be set using _____ . (GPIO.setmode())
20. The _____ function in Python allows you to read user input. (input)
21. The command to install the Adafruit DHT library is _____ . (git clone https://github.com/adafruit/Adafruit_Python_DHT.git)
22. The GPIO pin connected to the relay's input pin must be set as _____ in the code. (output)
23. The command to capture an image using the PiCamera is _____ . (raspistill -o image.jpg)
24. The Raspberry Pi camera module connects via the _____ slot. (CSI)
25. The _____ command is used to write data to a file in Python. (file.write())
26. The function to convert a string to an integer in Python is _____ . (int())
27. The _____ function is used to clear the GPIO settings in Python. (GPIO.cleanup())
28. The typical voltage for Raspberry Pi GPIO pins is _____ volts. (3.3)
29. To access a specific function from a module, you can use the _____ statement. (from...import)
30. The command to install the Pillow library is _____ . (sudo pip install pillow)
31. The Raspberry Pi can run various operating systems, including _____ . (Raspbian)
32. The GPIO pins on Raspberry Pi can act as both _____ and digital input/output. (analog)
33. The _____ command is used to check the current directory in the terminal. (pwd)
34. The Python keyword to define a loop is _____ . (for or while)
35. The _____ function is used to pause the execution of a program for a specified time. (time.sleep())

36. The command to list files in a directory is _____ . (ls)
37. The _____ function in Python returns the length of a list or string. (len())
38. To iterate over a list in Python, you can use a _____ loop. (for)
39. The command to update the package list on Raspberry Pi is _____ . (sudo apt-get update)
40. The _____ function is used to convert a list into a string in Python. (join())
41. The typical GPIO pin used for output in the LED blinking example is _____ . (11)
42. The Python keyword used to define a variable that cannot be changed is _____ . (const or final)
43. The command to reboot the Raspberry Pi is _____ . (sudo reboot)
44. The _____ method is used to read a single line from a file in Python. (readline())
45. The _____ function is used to check if a value is in a list in Python. (in)
46. The command to create a new directory in Linux is _____ . (mkdir)
47. The _____ function is used to return the maximum value from a list in Python. (max())
48. The command to install Python on Raspberry Pi is _____ . (sudo apt-get install python)
49. The _____ library is used for handling dates and times in Python. (datetime)
50. The command to check the Raspberry Pi's IP address is _____ . (hostname -I)
51. The _____ function is used to convert a string to uppercase in Python. (upper())
52. The command to install the RPi.GPIO library is _____ . (sudo pip install RPi.GPIO)
53. The _____ function is used to get the current date and time in Python. (datetime.now())
54. The command to list the contents of a directory in a long format is _____ . (ls -l)
55. The _____ function is used to convert a string to lowercase in Python. (lower())
56. The command to create a new file in Linux is _____ . (touch)
57. The _____ function is used to check if a file exists in Python. (os.path.exists())
58. The command to check the Raspberry Pi's CPU temperature is _____ . (vcgencmd measure_temp)
59. The _____ function is used to get the current working directory in Python. (os.getcwd())
60. The command to install the picamera library is _____ . (sudo pip install picamera)
61. The _____ function is used to convert a list into a tuple in Python. (tuple())
62. The command to check the Raspberry Pi's memory usage is _____ . (free -h)
63. The _____ function is used to get the current system time in Python. (time.time())
64. The command to install the Adafruit SSD1306 library is _____ . (sudo pip install Adafruit-SSD1306)
65. The _____ function is used to convert a tuple into a list in Python. (list())
66. The command to check the Raspberry Pi's disk usage is _____ . (df -h)

67. The _____ function is used to get the current system date in Python. (datetime.date.today())
68. The command to install the Adafruit DHT library is _____ . (sudo pip install Adafruit-DHT)
69. The _____ function is used to convert a string to a float in Python. (float())
70. The command to check the Raspberry Pi's system information is _____ . (uname -a)
71. The _____ function is used to get the current system time in seconds since the epoch in Python. (time.time())
72. The command to install the Adafruit BME280 library is _____ . (sudo pip install Adafruit-BME280)
73. The _____ function is used to convert a string to an integer in Python. (int())
74. The command to check the Raspberry Pi's network configuration is _____ . (ifconfig)
75. The _____ function is used to get the current system date and time in Python. (datetime.datetime.now())
76. The command to install the Adafruit TCS34725 library is _____ . (sudo pip install Adafruit-TCS34725)
77. The _____ function is used to convert a list into a set in Python. (set())
78. The command to check the Raspberry Pi's system logs is _____ . (sudo journalctl -u systemd)
79. The _____ function is used to get the current system time zone in Python. (time.timezone)
80. The command to install the Adafruit ADS1x15 library is _____ . (sudo pip install Adafruit-ADS1x15)
81. The _____ function is used to convert a string to a boolean in Python. (bool())
82. The command to check the Raspberry Pi's system services is _____ . (sudo systemctl status)
83. The _____ function is used to get the current system time in milliseconds since the epoch in Python. (int(time.time() * 1000))
84. The command to install the Adafruit INA219 library is _____ . (sudo pip install Adafruit-INA219)
85. The _____ function is used to convert a list into a dictionary in Python. (dict())
86. The command to check the Raspberry Pi's system environment variables is _____ . (printenv)
87. The _____ function is used to get the current system time in microseconds since the epoch in Python. (int(time.time() * 1000000))
88. The command to install the Adafruit MCP3008 library is _____ . (sudo pip install Adafruit-MCP3008)

89. The _____ function is used to convert a string to a hexadecimal in Python. (hex())
90. The command to check the Raspberry Pi's system hardware information is _____ . (lshw)
91. The _____ function is used to get the current system time in nanoseconds since the epoch in Python. (int(time.time() * 1000000000))
92. The command to install the Adafruit PCA9685 library is _____ . (sudo pip install Adafruit-PCA9685)
93. The _____ function is used to convert a list into a frozenset in Python. (frozenset())
94. The command to check the Raspberry Pi's system kernel version is _____ . (uname -r)
95. The _____ function is used to get the current system time in seconds since the epoch in Python. (time.time())
96. The command to install the Adafruit SHT31D library is _____ . (sudo pip install Adafruit-SHT31D)
97. The _____ function is used to convert a string to a binary in Python. (bin())
98. The command to check the Raspberry Pi's system CPU architecture is _____ . (uname -m)
99. The _____ function is used to get the current system time in milliseconds since the epoch in Python. (int(time.time() * 1000))
100. The command to install the Adafruit BMP280 library is _____ . (sudo pip install Adafruit-BMP280)

Week-7

1. The Internet of Things (IoT) is a network of _____ connected devices that interact with each other. (interconnected)
2. The main function of IoT is to create an _____ environment. (interactive)
3. Remote data logging involves collecting data from devices and sending it to a _____ . (server)
4. A common sensor used in IoT applications for temperature and humidity is the _____ sensor. (DHT)
5. The DHT sensor has four pins: Power, Data, Null, and _____. (Ground)
6. In the DHT sensor, Pin 1 is used for _____ supply. (power)
7. The Raspberry Pi GPIO pin used to connect the DHT sensor data pin is typically pin _____. (11)
8. To read data from the DHT sensor in Python, you can use the function _____ . (Adafruit_DHT.read_retry())

9. Socket programming allows for _____ communication between a client and a server. (two-way)
10. In socket programming, the command to create a socket is _____ . (socket.socket())
11. The server binds to a specific _____ to listen for client connections. (port)
12. The Python command to accept a client connection is _____ . (s.accept())
13. The client sends data to the server using the _____ method. (sendto)
14. To split a string in Python, you can use the _____ function. (split())
15. The library used for plotting data in Python is called _____. (Matplotlib)
16. The command to plot data in Matplotlib is _____ . (plot())
17. The basic architecture of Software-Defined Networking (SDN) includes an application layer, control layer, and _____ layer. (infrastructure)
18. SDN separates the _____ logic from hardware switches. (control)
19. The protocol used for communication between the control plane and data plane in SDN is called _____. (OpenFlow)
20. Flow-rules are maintained in a _____ table at each switch in SDN. (flow)
21. The challenge of _____ placement involves deciding where to place controllers in the network. (controller)
22. In SDN, a _____ controller can take over if the primary controller fails. (backup)
23. The _____ API is used to communicate between the control layer and infrastructure layer in SDN. (southbound)
24. The _____ API is used to communicate between the control layer and application layer in SDN. (northbound)
25. One of the benefits of integrating SDN in IoT is _____ routing decisions. (intelligent)
26. Wireless Sensor Networks (WSN) face challenges such as limited _____ and resource constraints. (memory)
27. The concept of _____-centric data forwarding involves sending data based on the value measured by a sensor. (value)
28. In Software-Defined WSN, the _____ can be programmed in real-time. (sensor nodes)
29. The command to create a UDP socket in Python is _____ . (socket.socket(socket.AF_INET, socket.SOCK_DGRAM))
30. The function used to read humidity and temperature from a DHT sensor is _____ . (Adafruit_DHT.read_retry(sensor, pin))
31. To save data received from a client, the server can write to a _____ file. (text)
32. A typical DHT sensor requires a _____ ohm resistor for proper operation. (4.7K)
33. The command to close a socket in Python is _____ . (sock.close())
34. The _____ function in Python is used to read user input. (input())

35. In socket programming, the server listens for connections using the command _____ . (s.listen())
36. The DHT sensor's data pin is connected to a _____ pin on the Raspberry Pi. (GPIO)
37. The command to plot the x-axis label in Matplotlib is _____ . (xlabel())
38. In a client-server model, the _____ performs the task requested by the client. (server)
39. The _____ command is used to bind a socket to a specific address and port. (bind())
40. In SDN, the _____ placement problem involves minimizing delays when defining rules. (rule)
41. The _____ timeout deletes flow rules after a set period. (hard)
42. The _____ timeout deletes flow rules if no traffic is received for a specified time. (soft)
43. To visualize data in real-time, you can use the _____ function in Matplotlib. (ion())
44. The data sent from the client to the server is typically formatted as a _____ . (string)
45. The command to read a single line from a file in Python is _____ . (readline())
46. The _____ library is used for creating plots in Python. (matplotlib.pyplot)
47. The _____ function is used to plot data in real-time. (plot())
48. The _____ command is used to set the title of a plot in Matplotlib. (title())
49. In SDN, the _____ API is used for communication among multiple controllers. (east-westbound)
50. The _____ protocol is used for communication between the control plane and data plane in SDN. (OpenFlow)
51. The _____ function is used to read data from a DHT sensor. (Adafruit_DHT.read())
52. The _____ command is used to send data from the client to the server. (sendto())
53. The _____ function is used to split a string into multiple strings. (split())
54. The _____ library is used for socket programming in Python. (socket)
55. The _____ command is used to create a socket in Python. (socket.socket())
56. The _____ function is used to read data from a file in Python. (read())
57. The _____ command is used to write data to a file in Python. (write())
58. The _____ function is used to close a file in Python. (close())
59. The _____ command is used to bind a socket to a specific address and port. (bind())
60. The _____ function is used to listen for connections in Python. (listen())
61. The _____ command is used to accept a client connection in Python. (accept())
62. The _____ function is used to send data from the client to the server. (sendto())
63. The _____ command is used to close a socket in Python. (close())

64. The _____ function is used to read data from a DHT sensor. (Adafruit_DHT.read())
65. The _____ library is used for plotting data in Python. (matplotlib.pyplot)
66. The _____ function is used to plot data in real-time. (plot())
67. The _____ command is used to set the title of a plot in Matplotlib. (title())
68. The _____ function is used to read data from a file in Python. (read())
69. The _____ command is used to write data to a file in Python. (write())
70. The _____ function is used to close a file in Python. (close())
71. The _____ command is used to bind a socket to a specific address and port. (bind())
72. The _____ function is used to listen for connections in Python. (listen())
73. The _____ command is used to accept a client connection in Python. (accept())
74. The _____ function is used to send data from the client to the server. (sendto())
75. The _____ command is used to close a socket in Python. (close())
76. The _____ function is used to read data from a DHT sensor. (Adafruit_DHT.read())
77. The _____ library is used for plotting data in Python. (matplotlib.pyplot)
78. The _____ function is used to plot data in real-time. (plot())
79. The _____ command is used to set the title of a plot in Matplotlib. (title())
80. The _____ function is used to read data from a file in Python. (read())
81. The _____ command is used to write data to a file in Python. (write())
82. The _____ function is used to close a file in Python. (close())
83. The _____ command is used to bind a socket to a specific address and port. (bind())
84. The _____ function is used to listen for connections in Python. (listen())
85. The _____ command is used to accept a client connection in Python. (accept())
86. The _____ function is used to send data from the client to the server. (sendto())
87. The _____ command is used to close a socket in Python. (close())
88. The _____ function is used to read data from a DHT sensor. (Adafruit_DHT.read())
89. The _____ library is used for plotting data in Python. (matplotlib.pyplot)
90. The _____ function is used to plot data in real-time. (plot())
91. The _____ command is used to set the title of a plot in Matplotlib. (title())
92. The _____ function is used to read data from a file in Python. (read())
93. The _____ command is used to write data to a file in Python. (write())
94. The _____ function is used to close a file in Python. (close())
95. The _____ command is used to bind a socket to a specific address and port. (bind())
96. The _____ function is used to listen for connections in Python. (listen())
97. The _____ command is used to accept a client connection in Python. (accept())

98. The _____ function is used to send data from the client to the server. (sendto())
99. The _____ command is used to close a socket in Python. (close())
100. The _____ function is used to read data from a DHT sensor. (Adafruit_DHT.read())