Internal Assessment Test II – Dec. 2025

| Subject/Code: Internet of Things/ BCS701 | | | | |
|---|---|---|---|---|
| Date: 01/12/25 | Duration: 90 mins | Max. Marks:50 | Semester: 07 | Branch: CSE |

| | **Answer any FIVE FULL Questions** | **Marks** | **CO** | **RBT** |
|---|---|---|---|---|
| 1a) | What are the steps involved in IOT system design methodology.<br><br>IoT Design Methodology that includes: <br><br>Purpose & Requirements Specification <br>Process Specification<br>Domain Model Specification<br>Information Model Specification<br>Service Specifications <br>IoT Level Specification<br>Functional View Specification <br>Operational View Specification<br>Device & Component Integration<br>Application Development<br><br>Explantation of each | 5 | CO3 | L2 |
| b) | Explain with an example domain model specification in detail.<br><br>The third step in IoT design methodology is to define domain model for an application. Domain model should describe the main concepts, entities and objects in the domain of IoT system to be designed. Domain model should also defines the attribute of objects an relationship between objects.d. The domain model also provides the abstract representation of the concepts, objects and entities in the IoT domain independent of any specific technology or platform. With the domain model the IoT designers can get an understanding of the IoT domain for which system is to be designed.<br><br>Domain model specification (smart home automation) 1. Physical Entity (e.g. room, car etc) 2. Virtual Entity 3. Devices (physical things) 4. Resources 5. service | 5 | CO3 | L2 |
| 2a) | What is string concatenation in python explain with one example.Mention diffence between lists and tuples in python.<br><br>String concatenation in Python is the process of joining two or more strings into a single, longer string. Python offers several methods, with the most common ones being the + operator, the .join() method, and f-strings.<br><br>Example<br>str1 = "Hello"str2 = "World"result = str1 + " " + str2<br>print(result)# Output: Hello World<br><br>The primary difference between Python lists and tuples is mutability: lists are mutable (changeable), while tuples are immutable (unchangeable). This core distinction impacts their syntax, performance, and use cases.<br><br>Use a list when you have a collection of items that needs to be dynamically modified (added, removed, or changed) during program execution, such as a shopping cart or a list of user inputs.<br><br>Use a tuple when you have a fixed collection of data that should not change, ensuring data integrity. Common examples include geographic coordinates (latitude, longitude), RGB color values, or database records that are returned from a function. The immutability also offers a slight performance and memory benefit. | 6 | CO3 | L2 |

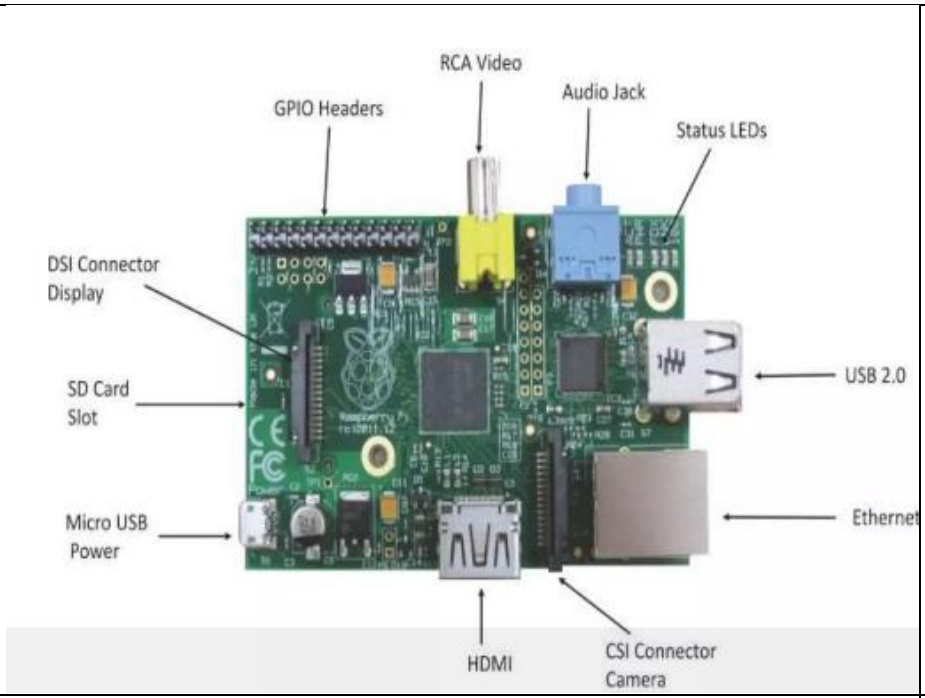
| | | | | |
|---|---|---|---|---|
| b) | How do you create an empty dictionary? Explain the difference between dict.keys(), dict.values(), and dict.items()<br><br>Dictionary<br><br>A dictionary object is an unordered collection of data in a key:value pair form. A collection of such pairs is enclosed in curly brackets.<br><br># Creating an empty dictionary<br>empty_dict = {}<br><br>keys()<br>A key is the unique identifier used to access a specific value in a dictionary, similar to an index in a list. Keys must be immutable (e.g., strings, numbers, or tuples).<br>The .keys() method returns a view object of all the keys in the dictionary<br>my_dict = {'apple': 3, 'banana': 5, 'cherry': 7}<br>print(my_dict.keys())<br># Output: dict_keys(['apple', 'banana', 'cherry'])<br><br>for key in my_dict.keys():<br>print(key)<br><br>values()<br>A value is the data associated with a key. Values can be of any data type and can contain duplicates, unlike keys.<br>The .values() method returns a view object of all the values in the dictionary.<br>my_dict = {'apple': 3, 'banana': 5, 'cherry': 7}<br>print(my_dict.values())<br># Output: dict_values([3, 5, 7])<br><br>for value in my_dict.values():<br>print(value)<br><br>items()<br>An item refers to the entire key-value pair as a single entity, represented as a tuple (key, value).<br>The .items() method returns a view object of all the items (key-value tuples) in the dictionary.<br>my_dict = {'apple': 3, 'banana': 5, 'cherry': 7}<br>print(my_dict.items())<br># Output: dict_items([('apple', 3), ('banana', 5), ('cherry', 7)])<br><br>for key, value in my_dict.items():<br>print(f"Key: {key}, Value: {value}") | 4 | CO3 | L3 |
| 3a) | What is a keyword argument in python? What are variable length arguments?<br><br>A keyword argument in Python is an argument passed to a function call that is preceded by a named parameter and the assignment operator (=), explicitly linking the value to a specific parameter name<br>Using keyword arguments makes function calls more readable and allows for flexibility in argument order<br><br>def greet(name, message):<br>return f"Hello, {name}! {message}"<br># Using keyword arguments (order doesn't matter)<br>print(greet(message="Good afternoon!", name="Bob"))<br><br>Variable Length Arguments<br>Variable length arguments (or variadic arguments) allow a function to accept an arbitrary number of arguments .<br><br>Explain with example | 4 | CO3 | L2 |

| | | | | |
|---|---|---|---|---|
| b) | What is the difference between a python module and a package?How is function overriding implemented in python?<br><br>A Python module is a single file containing Python code, while a package is a directory that organizes related modules into a hierarchy.<br><br>**Module** / **Package**<br><br>A single `.py` file. / A directory containing one or more modules and sub-packages.<br><br>To organize related functions, classes, and variables into a single file. / To organize related modules into a logical hierarchy and prevent naming collisions.<br><br>Named after the filename (e.g., `my_module.py` is the module `my_module`). / Identified by the presence of a special `__init__.py` file (in Python 3.3+ this file is optional, but common).<br><br>Imported using `import module_name` or `from module_name import ...`. / Imported using dot notation to navigate th hierarchy, e.g., `import package.module`.<br><br>.<br>Function overriding in Python is a core feature of object-oriented programming (OOP) achieved by defining a method in a subclass with the exact same name and signature (parameters) as a method in its superclass. Python's dynamic method resolution ensures the subclass's method is executed at runtime when called on an instance of the subclass.<br><br>class Shape:<br>  def no_of_sides(self):<br>  print("This shape has many sides")<br><br>class Square(Shape):<br>  def no_of_sides(self):<br>  print("A square has 4 sides")<br><br># Creating objects<br>s1 = Shape()<br>s2 = Square()<br><br># Calling the method<br>s1.no_of_sides()  # Parent class method<br>s2.no_of_sides()  # Child class overrides the method | 6 | CO3 | L3 |

| | | | | |
|---|---|---|---|---|
| 4.a) | Explain Raspberry Pi board with the various componnets /peripherals. | 7 | CO4 | L2 |

- Raspberry Pi is a low-cost mini-computer with the physical size of a credit card.
- Raspberry Pi runs various flavors of Linux and can perform almost all tasks that a normal desktop computer can do.
- Raspberry Pi also allows interfacing sensors and actuators through the general purpose I/O pins.
- Since Raspberry Pi runs Linux operating system, it supports Python "out of the box"

- **Processor & RAM** : Raspberry Pi is based on an ARM processor. The latest version of Raspberry Pi (Model B, Revision 2) comes with 700 MHz Low Power ARM1176JZ-F processor and 512 MB SDRAM.
- **USB Ports** : Raspberry Pi comes with two USB 2.0 ports. The USB ports on Raspberry Pi can provide a current upto 100mA. For connecting devices that draw current more than 100mA, an external USB powered hub is required.
- **Ethernet Ports** : Raspberry Pi comes with a standard RJ45 Ethernet port. You can connect an Ethernet cable or a USB Wifi adapter to provide Internet connectivity.
- **HDMI Output** : The HDMI port on Raspberry Pi provides both video and audio output. You can connect the Raspberry Pi to a monitor using an HDMI cable. For monitors that have a DVI port but no HDMI port, you can use an HDMI to DVI adapter/cable.
- **Composite Video Output** : Raspberry Pi comes with a composite video output with an RCA jack that supports both PAL and NTSC video output. The RCA jack can be used to connect old televisions that have an RCA input only.
- **Audio Output** : Raspberry Pi has a 3.5mm audio output jack. This audio jack is used for providing audio output to old televisions along with the RCA jack for video. The audio quality from this jack is inferior to the HDMI output.
- **GPIO Pins** : Raspberry Pi comes with a number of general purpose input/ouput pins. Figure 7.3 shows the Raspberry Pi GPIO headers. There are four types of pins on Raspberry Pi - true GPIO pins, I2C interface pins, SPI interface pins and serial Rx and Tx pins.
- **Display Serial Interface (DSI)** : The DSI interface can be used to connect an LCD panel to Raspberry Pi.
- **Camera Serial Interface (CSI)** : The CSI interface can be used to connect a camera module to Raspberry Pi.
- **Status LEDs** : Raspberry Pi has five status LEDs. Table 7.1 lists Raspberry Pi status LEDs and their functions.
- **SD Card Slot** : Raspberry Pi does not have a built in operating system and storage. You can plug-in an SD card loaded with a Linux image to the SD card slot. Appendix-A provides instructions on setting up New Out-of-the-Box Software (NOOBS) on Raspberry Pi. You will require atleast an 8GB SD card for setting up NOOBS.
- **Power Input** : Raspberry Pi has a micro-USB connector for power input.

| | | | | |
|---|---|---|---|---|
| |  | | | |
| b) | How is Raspberry Pi different from a desktop computer?<br><br>A Raspberry Pi is a small, low-power, single-board computer that is less powerful than a desktop PC, lacks modular components like a replaceable CPU, and often uses a microSD card for storage. In contrast, a desktop computer is built from individual, modular components like a motherboard, CPU, RAM, and graphics card, is much more powerful, and typically has dedicated internal storage like an SSD or hard drive | 3 | CO4 | L2 |
| 5. | Explain how to design a home intrusion detection system and describe the steps involved in detecting unauthorized entry.<br><br>Explanation of different sensors like PIR Sensors etc and diagrams related to the system | 10 | CO4 | L2 |
| 6. | Explain how a Hadoop MapReduce job is executed, describing the key steps involved in the job execution workflow<br><br>Steps in MapReduce Job Execution<br>Step 1: Job Submission<br>Client submits job → JobTracker.<br>JobTracker returns a unique JobID to the client for tracking progress.<br>Step 2: Locate Input Data<br><br>JobTracker contacts NameNode to determine:<br>Locations of input data blocks in HDFS.<br>Which DataNodes store the required blocks.<br>Step 3: Task Assignment<br><br>JobTracker identifies TaskTracker nodes with free task slots.<br>Priority is given to nodes:<br>With data locality (same node as data)<br>Or at least same rack<br>Step 4: Heartbeat Mechanism<br><br>TaskTrackers send heartbeat messages to JobTracker every few minutes.<br>Heartbeat message includes:<br>Confirmation that TaskTracker is alive<br>Number of free task slots<br>Task execution status<br>JobTracker uses this info to assign new tasks. | 10 | CO5 | L2 |

Step 5: Scheduling of Tasks

TaskTrackers poll JobTracker for tasks.
JobTracker assigns tasks using scheduling algorithms.
Default Hadoop Scheduling Algorithm: FIFO
FIFO → First-In, First-Out.
Job queue is maintained.
Oldest job is executed first.
No consideration for:
Job size
Job priority

Step 6: Task Execution

Each TaskTracker:
Runs each task in a separate JVM process.
This ensures failure in one task does not crash the TaskTracker.
TaskTracker monitors:
Task output
Logs
Exit codes

Step 7: Task Completion Reporting

When a task completes (success or failure):
TaskTracker immediately notifies JobTracker.

Step 8: Handling Task Failures

If a task fails:
TaskTracker reports failure to JobTracker.
JobTracker decides to:
Re-execute task on another TaskTracker, or
Skip/mark record if repeatedly failing
If a TaskTracker has repeated failures:
JobTracker blacklists TaskTracker as unreliable.
Future tasks are not assigned to it.