

Boost up Your Certification Score

WGU

Foundations-of-Computer-Science

WGU Foundations of Computer Science



For More Information – Visit link below:

<https://www.examsboost.com/>

Product Version

- ✓ Up to Date products, reliable and verified.
- ✓ Questions and Answers in PDF Format.

Latest Version: 6.0

Question: 1

What type of encryption is provided by encryption utilities built into the file system?

- A. Encryption in motion
- B. Encryption authentication
- C. Encryption at rest
- D. Encryption steganography

Answer: C

Explanation:

File system encryption utilities are designed to protect data stored on a disk—for example, files on an SSD, HDD, or other persistent storage. This protection is called encryption at rest. The key idea is that if an attacker steals the physical drive, gains access to a powered-off machine, or otherwise reads storage directly, the raw bytes on disk remain unreadable without the correct cryptographic key. Common textbook examples include full-disk encryption and per-file encryption supported by operating systems and file systems.

This differs from encryption in motion (also called encryption in transit), which protects data while it is being transmitted over networks, such as via TLS/HTTPS, VPNs, or secure messaging protocols. File system utilities do not primarily address network transmission; they address stored data confidentiality. Option B, “encryption authentication,” is not a standard category; authentication is a security goal often achieved using mechanisms like digital signatures, MACs, certificates, and protocol handshakes, not a type of file system encryption. Option D, steganography, is the practice of hiding information within other data (like images or audio) rather than encrypting it for confidentiality.

In short, file system encryption utilities aim to ensure that stored files remain confidential if storage is accessed without authorization, which is precisely the definition of encryption at rest.

Question: 2

Which Windows 11 tool enables a user to manually add a Bluetooth device if it does not automatically configure when first connected?

- A. Network center
- B. Task scheduler
- C. Windows defender
- D. Device manager

Answer: D

Explanation:

When a Bluetooth device does not configure automatically, the underlying issue is often driver discovery, device enumeration, or the Bluetooth adapter's state. In Windows, the tool traditionally associated with manually managing hardware devices and their drivers is Device Manager . It lets a user view hardware categories (including Bluetooth adapters), enable or disable devices, update drivers, uninstall and rescan, and address "unknown device" situations. These actions are core to manual configuration because they influence whether Windows can properly recognize and communicate with a Bluetooth device.

Windows 11 pairing itself is typically initiated from the Settings app under Bluetooth and devices, where a user chooses "Add device" to pair a new accessory. (Microsoft Support) However, among the options provided, only Device Manager is a hardware-configuration tool that can resolve situations where automatic configuration fails due to driver or adapter problems. Network-related tools do not handle local device drivers, Task Scheduler automates tasks rather than adding devices, and Windows Defender is focused on security and malware protection rather than device setup.

From a systems perspective, this reflects a key operating-systems concept: successful device use requires both discovery/pairing and a correctly installed driver stack. Device Manager is the standard interface for the driver and device side of that equation, which is why it is the best match to "manually add or configure" hardware in the given choices.

Question: 3

What is the method for changing an element in a Python list?

- A. Use square brackets and the equals sign
- B. Use parentheses and the plus sign
- C. Use curly brackets and the equals sign
- D. Use the del keyword and the element's value

Answer: A

Explanation:

In Python, a list is a mutable sequence, meaning its elements can be changed after the list is created. The standard textbook method for updating a specific element is index assignment , which uses square brackets to select the position and the equals sign to assign a new value. For example, if `nums = [10, 20, 30]`, then `nums[1] = 99` changes the element at index 1 from 20 to 99, producing `[10, 99, 30]`. This works because lists store references to objects and allow those references to be updated in-place.

Option B is incorrect because parentheses are used for function calls and tuples, and the plus sign typically performs concatenation (creating a new list) rather than modifying an existing element by position. Option C is incorrect because curly brackets denote dictionaries or sets, not lists. Option D is incorrect because `del` removes elements by index or slice (for example, `del nums[1]`), and it does not delete by "the element's value" unless you first find the index. Deleting is not the same as changing; deletion reduces the list's length and shifts later indices.

Index assignment is fundamental in list manipulation and appears in standard algorithms: updating counters, replacing sentinel values, editing collections, and implementing in-place transformations efficiently without allocating a new list.

Question: 4

What is the expected output of calling `.shape` on a NumPy 2D array?

- A. The type of elements in the array
- B. The number of rows and columns in the 2D array
- C. The sum of the dimensions of the array
- D. The total number of elements in the array

Answer: B

Explanation:

In NumPy, every ndarray has a shape attribute that describes the size of the array along each dimension. For a 2D array, shape returns a tuple with two integers: (number_of_rows, number_of_columns). For example, if `a = np.array([[1, 2, 3], [4, 5, 6]])`, then `a.shape` is (2, 3), meaning 2 rows and 3 columns. This is a fundamental idea in matrix and array computing, because shape governs how indexing, slicing, broadcasting, and linear algebra operations behave.

Option A describes the dtype, which can be accessed with `a.dtype`, not `a.shape`. Option C is incorrect because shape provides per-dimension sizes, not their sum. Option D refers to the total number of elements, which NumPy provides via `a.size` (or equivalently `np.prod(a.shape)`).

Textbooks emphasize shape because many errors in numerical computing come from mismatched dimensions. For example, matrix multiplication requires compatible inner dimensions, and broadcasting rules depend on dimension sizes. By checking `.shape`, programmers can verify their data layout before applying algorithms, ensuring rows represent observations and columns represent features (or vice versa). Thus, for a 2D NumPy array, `.shape` indicates the number of rows and columns.

Question: 5

Which sorting algorithm works by finding the smallest or largest element in an unsorted part of a list and moving it to the sorted part of the list?

- A. Radix sort
- B. Heap sort
- C. Quicksort
- D. Selection sort

Answer: D

Explanation:

Selection sort is defined by a simple repeated strategy: divide the list into a sorted region and an unsorted region, then repeatedly select the smallest (or largest) element from the unsorted region and move it to the end of the sorted region. In the common “smallest-first” version, the algorithm scans the unsorted portion to find the minimum element, then swaps it into the next position in the sorted portion. After the first pass, the smallest element is fixed at index 0; after the second pass, the second-smallest is fixed at index 1; and so on until the entire list is sorted.

This exactly matches the description in the question, making selection sort the correct answer.

Textbooks often use selection sort to teach algorithmic thinking because it is easy to understand and

implement, though not efficient for large datasets. Its time complexity is $O(n^2)$ in the average and worst case because it performs roughly n scans of progressively smaller unsorted sections, with each scan taking linear time. Its space usage is $O(1)$ additional space because it sorts in place using swaps. The other options do not match the described mechanism. Quicksort partitions around a pivot, heap sort uses a heap data structure to repeatedly extract the maximum/minimum, and radix sort processes digits/keys by place value rather than selecting minima by scanning. Selection sort's defining action is the repeated "select the min/max and place it."

Question: 6

Which aspect is excluded from a NumPy array's structure?

- A. The data pointer
- B. The shape of the array
- C. The data type or dtype pointer
- D. The encryption key of the array

Answer: D

Explanation:

A NumPy ndarray is designed for efficient numerical computing, and its structure is defined by metadata required to interpret a contiguous (or strided) block of memory as an n -dimensional array. Textbooks and NumPy's own conceptual model describe key components such as: a data buffer (where the raw bytes live), a data pointer (reference to the start of that buffer), the dtype (which specifies how to interpret each element's bytes—e.g., int32, float64), the shape (the size in each dimension), and strides (how many bytes to step in memory to move along each dimension). Together, these allow fast indexing, slicing, and vectorized operations without Python-level loops.

Options A, B, and C are all part of what an array must track to function correctly: the array must know where its data is, how it is laid out (shape/strides), and how to interpret bytes (dtype). In contrast, an encryption key is not a concept that belongs to the internal representation of a numerical array.

Encryption is a security mechanism applied at storage or transport layers (for example, encrypting a file on disk or encrypting data sent over a network), not something built into the in-memory structure of a NumPy array object.

Therefore, the aspect excluded from a NumPy array's structure is the encryption key.

Question: 7

What is the layer of programming between the operating system and the hardware that allows the operating system to interact with it in a more independent and generalized manner?

- A. The hardware abstraction layer
- B. The boot loader layer
- C. The task scheduler layer
- D. The file system layer

Answer: A

Explanation:

The Hardware Abstraction Layer (HAL) is a software layer that sits between the operating system kernel and the physical hardware. Its purpose is to hide hardware-specific details behind a consistent interface, allowing the OS to be more portable and easier to maintain across different hardware platforms.

Textbooks explain that without abstraction, the OS would need extensive device- and architecture-specific code scattered throughout the kernel, making updates and cross-platform support far more difficult.

The HAL typically provides standardized functions for interacting with low-level components such as interrupts, timers, memory mapping, and device I/O. With a HAL, the OS can call general routines (for example, to configure an interrupt controller) while the HAL handles the platform-specific implementation. This supports a key systems principle: separate policy (what the OS wants to do) from mechanism (how hardware accomplishes it).

The other options are not correct. A boot loader runs at startup to load the operating system into memory; it is not the general interface layer during normal operation. The task scheduler is a kernel subsystem that manages CPU time among processes, not a hardware-independence layer. The file system layer manages storage organization and access semantics; it is not the general abstraction for all hardware interactions.

Therefore, the programming layer that enables generalized OS interaction with hardware is the hardware abstraction layer.

Question: 8

What Python code would return the value 40 from np_2d, where np_2d = np.array([[1, 2, 3, 4], [10, 20, 30, 40]])?

- A. np_2d[3, 1]
- B. np_2d[1, 3]
- C. np_2d[0, 4]
- D. np_2d[4, 1]

Answer: B

Explanation:

In a 2D NumPy array, indexing is written as array[row_index, column_index] using zero-based indices.

The array np_2d = np.array([[1, 2, 3, 4], [10, 20, 30, 40]]) has two rows (indices 0 and 1) and four columns (indices 0, 1, 2, 3). The value 40 is located in the second row and the fourth column. Using zero-based indexing, that corresponds to row index 1 and column index 3. Therefore, np_2d[1, 3] returns 40.

Option A attempts to access row 3, which does not exist and would raise an IndexError. Option C attempts to access column 4 in row 0, but valid column indices are only 0 through 3, so it would also error. Option D likewise refers to a non-existent row 4. Only option B uses valid indices and points to the correct location.

Textbooks emphasize multi-dimensional indexing because it underlies matrix operations, dataset manipulation, and feature extraction in data science. Correctly interpreting rows and columns is essential when rows represent observations (like people) and columns represent attributes (like age, weight, height). This question tests precise control over row/column addressing, which prevents subtle bugs in numerical analysis.

Question: 9

Which type of files are meant to be inaccessible to standard users, but can be critical in terms of functionality?

- A. Extension files
- B. Log files
- C. System files
- D. Backup files

Answer: C

Explanation:

Operating systems contain many files that are essential for booting, hardware support, security enforcement, and core services. These are generally referred to as system files. Textbooks explain that system files are often protected by permissions and special attributes because accidental modification or deletion could destabilize the OS, break device drivers, prevent applications from running, or even stop the machine from booting. Therefore, standard (non-administrator) users are typically restricted from accessing or altering them, and the OS may hide them by default to reduce the risk of user error. Examples include kernel-related components, shared libraries, driver files, configuration databases, and critical service executables. Modern OS designs enforce protection through user accounts, access control lists, and privilege separation. This ensures only trusted processes and administrators can change system-critical components.

Log files record events and are sometimes protected, but many logs are readable by users or administrators depending on policy; they are not necessarily “meant to be inaccessible” in the same strict sense. Backup files are important for recovery but are not inherently system-critical for day-to-day operation, and their accessibility depends on organizational policy. “Extension files” is not a standard category; file extensions describe formats rather than a protected functional class.

Thus, the files intended to be inaccessible to standard users yet critical for functionality are system files, reflecting core OS security principles such as least privilege and integrity protection.

Question: 10

Which protocol provides encryption while email messages are in transit?

- A. FTP
- B. HTTP
- C. TLS
- D. IMAP

Answer: C

Explanation:

“Encryption in transit” means protecting data while it moves across a network so that eavesdroppers cannot read or modify it. For email systems, this protection is most commonly provided by TLS

(Transport Layer Security) . TLS is a cryptographic protocol that can wrap application protocols (including mail protocols) to provide confidentiality, integrity, and server (and sometimes client) authentication. In practice, TLS is used to secure connections such as SMTP submission (often with STARTTLS or implicit TLS), IMAP over TLS, and POP3 over TLS. Textbooks present TLS as the standard successor to SSL and the foundation of secure communication on the modern Internet.

The other options are not correct in this context. FTP is a file transfer protocol and is traditionally unencrypted unless paired with additional security mechanisms (e.g., FTPS, which uses TLS, or SFTP, which uses SSH). HTTP is a web protocol; it becomes encrypted only when used as HTTPS, which again relies on TLS underneath. IMAP is an email retrieval protocol, but IMAP itself is not the encryption protocol —IMAP can be run over TLS (IMAPS) to become secure.

Therefore, the protocol that provides encryption while email messages (or email protocol traffic) are in transit is TLS.

Thank You for Trying Our Product

For More Information – **Visit link below:**

<https://www.examsboost.com/>

15 USD Discount Coupon Code:

G74JA8UF

FEATURES

- ✓ **90 Days Free Updates**
- ✓ **Money Back Pass Guarantee**
- ✓ **Instant Download or Email Attachment**
- ✓ **24/7 Live Chat Support**
- ✓ **PDF file could be used at any Platform**
- ✓ **50,000 Happy Customer**



Visit us at: <https://www.examsboost.com/test/foundations-of-computer-science>