Q: Яка головна мета розбиття кодової бази проєкту на окремі файли, кожен з яких описує окремий функціонал?

a) Зменшення загального розміру коду.

b) Збільшення швидкості виконання програми.

c) Модульність коду для кращої організації, обслуговування та управління залежностями.

d) Використання різних мов програмування.

A: 3

Q: Яка перевага модульного коду полягає у створенні окремої області видимості для змінних, що запобігає конфліктам імен?

a) Підтримка

b) Повторне використання

c) Просторова оптимізація

d) Простір імен

A: 4

Q: Який сучасний стандарт системи модулів JavaScript відповідає за організацію та уніфікований підхід до імпорту та експорту даних між файлами?

a) CommonJS

b) AMD

c) ECMAScript Modules (ESM)

d) UMD

A: 3

Q: Що таке модуль в контексті ECMAScript Modules (ESM)?

a) HTML-файл з JavaScript-кодом.

b) CSS-файл з правилами стилів.

c) JavaScript-файл із фрагментом коду, який можна використовувати багато разів, експортуючи певні об'єкти.

d) Файл конфігурації проєкту.

A: 3

Q: Яка конструкція використовується для реалізації експорту за замовчуванням (default export) в ESM?

a) export { name }

b) import name from "..."

c) export default exportedValue

d) import { name as newName } }

A: 3

Q: Скільки експортів за замовчуванням (default export) може бути в одному модулі ESM?

a) Необмежена кількість.

b) Тільки один.

c) Два.

d) Не більше десяти.

A: 2

Q: Яка конструкція використовується для реалізації імпорту за замовчуванням (default import) в ESM?

a) export default ...

b) import name from "..."

c) import { name } from "..."

d) export { name as newName }

A: 2

Q: Чи повинно ім'я, з яким імпортується значення за замовчуванням, збігатися з іменем, яке було використано при експорті за замовчуванням?

a) Так, завжди.

b) Ні, ім'я може бути вибрано довільним чином.

c) Тільки якщо це функція.

d) Тільки якщо це змінна.

A: 2

Q: Який спосіб експортування дозволяє експортувати одне або кілька значень з модуля під конкретними, вказаними іменами?

a) Експорт за замовчуванням (default export).

b) Іменований експорт (named export).

c) Імпорт простору імен (namespace import).

d) Динамічний імпорт.

A: 2

Q: Яка конструкція використовується для реалізації іменованого експорту (named export) в ESM?

a) export default ...

b) import { name } from "..."

c) export exportedValue (наприклад, export const name = ...;)

d) import name from "..."

A: 3

Q: Чи обмежена кількість іменованих експортів в одному модулі ESM?

a) Так, можна експортувати не більше одного.

b) Так, можна експортувати не більше десяти.

c) Ні, кількість іменованих експортів в одному модулі не обмежена.

d) Залежить від браузера.

A: 3

Q: Яка конструкція використовується для реалізації іменованого імпорту (named import) в ESM?

a) export { name }

b) import name from "..."

c) import { name } from "..."

d) export default ...

A: 3

Q: Чи повинно ім'я, з яким імпортується іменоване значення, збігатися з іменем, яке було використано при іменованому експорті?

a) Ні, ім'я може бути вибрано довільним чином.

b) Так, ім'я імпорту повинно збігатися з іменем експорту.

c) Тільки якщо імпортується функція.

d) Тільки якщо імпортується змінна.

A: 2

Q: Яке ключове слово використовується для перейменування іменованої сутності під час її імпорту з іншого модуля?

a) rename

b) as

c) alias

d) with

A: 2

Q: Яка техніка дозволяє імпортувати весь вміст модуля (всі іменовані експорти) як властивості одного об'єкта з певним іменем?

a) Експорт за замовчуванням.

b) Іменований імпорт.

c) Перейменування імпорту.

d) Імпорт простору імен (namespace import).

A: 4

Q: Яка конструкція використовується для реалізації імпорту простору імен (namespace import)?

a) import name from "..."

b) import { \* as name } from "..."

c) import \* as name from "..."

d) export \* as name

A: 3

Q: Якщо модуль експортує export const name = "Mango"; і ми використовуємо import \* as user from './module.js';, як ми отримаємо доступ до значення змінної name через імпортований простір імен?

a) name

b) user.name

c) user.default.name

d) user["name"]

A: 2

Q: Яка перевага імпорту простору імен з точки зору уникнення конфліктів імен?

a) Він автоматично перейменовує всі імпортовані сутності.

b) Всі іменовані експорти стають властивостями одного об'єкта, що дозволяє уникнути конфліктів з іменами в поточному модулі.

c) Він імпортує лише ті сутності, яких ще немає в поточному модулі.

d) Імпорт простору імен не впливає на конфлікти імен.

A: 2

Q: Чи імпортується експорт за замовчуванням (default export) автоматично при використанні імпорту простору імен import \* as name from '...'?

a) Так, він стає властивістю default об'єкта простору імен.

b) Ні, імпорт простору імен стосується лише іменованих експортів.

c) Так, він стає основною властивістю об'єкта простору імен.

d) Залежить від налаштувань експорту.

A: 2

Q: Яка з перерахованих переваг не стосується модульності коду відповідно до наданого матеріалу?

a) Краща підтримка та можливість розширення функціоналу.

b) Запобігання забрудненню глобального простору імен.

c) Складність тестування окремих частин коду.

d) Можливість повторного використання коду.

A: 3