1. **Що таке клас, об’єкт, екземпляр класу? Що таке властивість та конструктор? Які бувають види властивостей та конструкторів?**

Клас – це конструкція мови, яка складається з ключового слова class, ідентифікатора та тіла.

Об’єкт - є конкретним представником класу, який існує в пам'яті під час виконання програми. Містить в собі статичні поля та всі методи.

Екземпляр класу - це саме такий об'єкт. Містять нестатичні поля.

Властивість – це конструкція мови C#, яка заміняє собою використання звичайних методів доступу.

Конструктор – спец. Метод, який викликається під час побудови класу.

Види властивостей:

* Read-Write властивість - яка дозволяє як зчитувати, так і встановлювати значення.
* Read-Only властивість - яка дозволяє тільки зчитувати значення, але не встановлювати його.
* Write-Only властивість - яка дозволяє тільки встановлювати значення, але не зчитувати його.

Види конструкторів:

* Конструктор за замовчуванням – ініціалізація полів значеннями за замовченням.
* Конструктор користувача – ініціалізація полів зумовленими користувачем значенням

1. **Що таке ООП? Описати основні парадигми OOП. Що таке Cast, Upcast, Downcast? Що таке модифікатори доступу та де їх використовують?**

ООП – парадигма програмування, в якій основними концепціями є поняття класів та об’єктів.

Основні парадигми:

Абстракція – це надання об’єкту характеристик, які відрізняють його від всіх інших об’єктів, чітко визначаючи його концептуальні межі.

Наслідування – дозволяє описати новий клас на основі вже існуючого (батьківського), при цьому властивості і функціональність батьківського класу запозичуються новим класом.

Поліморфізм – можливість об’єктів з однаковою специфікацією мати різну реалізацію.

* Cast - використовується для зміни типу об'єкта в інший тип даних.
* Upcast - відбувається, коли посилання на похідний клас приводиться до базового класу.
* Downcast - відбувається, коли посилання на базовий клас приводиться до похідного класу.

Модифікатори доступу використовуються для контролю доступу до членів класу. Їх використовують для забезпечення безпеки та налагодження коду. Основні модифікатори доступу включають:

* public - доступний з будь-якої точки програми.
* private - доступний лише в межах класу.
* protected - доступний у межах класу та його похідних класів.
* internal - доступний у межах того самого збірника.
* protected internal - поєднання модифікаторів protected і internal.

Ці модифікатори використовуються для керування доступом до полів, властивостей, методів та інших членів класу.

1. **Що таке абстрактний клас та інтерфейс? В чому між ними різниця?**

Абстрактний клас – це базовий клас, що не передбачає створення екземплярів через виклик конструктора безпосередньо, але екземпляр абстрактного класу створюється неявно при побудові екземпляра похідного конкретного класу.

Інтерфейс – семантична і синтаксична конструкція в коді програми, яка використовується для специфікування послуг,

Основна різниця між абстрактним класом і інтерфейсом полягає в тому, що клас може успадковувати тільки один абстрактний клас, але він може реалізувати декілька інтерфейсів. Крім того, в абстрактному класі можуть бути реалізовані часткові методи, але в інтерфейсі - ні.

1. **Що таке структура? В чому різниця між класом та структурою?**

Структура – це конструкція мови, що дозволяє утримати в собі набір полів різних типів.

Основна різниця між класом і структурою полягає у таких моментах:

* Семантика передачі: Класи передаються за посиланням, тобто коли об'єкт класу передається у метод, це передає посилання на об'єкт, а не сам об'єкт. Структури передаються за значенням, що означає, що при передачі структури у метод передається копія самої структури.
* Використання пам'яті: Об'єкти класу зазвичай розміщуються в кучі, тоді як екземпляри структур зберігаються в стеку.
* Наслідування: Класи можуть успадковувати інші класи, але структури не можуть успадковуватися від інших структур або класів.
* Нульове значення: Екземпляри класу можуть мати значення null, тоді як екземпляри структури завжди мають значення за замовчуванням.

Класи зазвичай використовуються для складних об'єктів зі складним життєвим циклом, тоді як структури зазвичай використовуються для невеликих даних, які мають простий життєвий цикл.

1. **Що таке делегат? Що таке лямбда вираз та чим він відрізняється від лямбда оператора?**

Делегат – це різновид об’єктів, які містять в собі покажчики на методи.

Лямбда вираз – це анонімні методи, які містять вирази й оператори і може використовуватися для створення делегатів.

Лямбда оператор – це багатооператорний лямбда-вираз. А лямбда-вираз – це однооператорний лямбда-оператор.

1. **Що таке потік? Який клас необхідно використовувати для створення екземпляра потоку? Який делегат необхідно використовувати для передачі метода в потік?**

Потік - є шляхом виконання коду програми паралельно з іншими частинами коду. Він дозволяє виконувати декілька фрагментів коду одночасно, забезпечуючи багатозадачність в програмі. Потоки дозволяють виконувати операції в фоновому режимі, не блокуючи основний потік виконання програми.

Для створення екземпляра потоку в C# використовується клас Thread з простору імен System.Threading. Для цього ви маєте створити новий екземпляр класу Thread і передати йому метод, який потрібно виконати в новому потоці.

Для передачі метода в потік необхідно використовувати делегат ThreadStart для методів без параметрів або ParameterizedThreadStart для методів з параметрами.