1. \*\*What is a delegate in C#?\*\*

- A) A class for managing arrays

- B) A reference type that defines a method signature

- C) A type used for handling exceptions

- D) A keyword for data access

\*\*Answer: B) A reference type that defines a method signature\*\*

2. \*\*How do you declare a delegate in C#?\*\*

- A) `delegate returnType delegateName(parameters);`

- B) `delegate returnType delegateName(parameters) { }`

- C) `delegate returnType delegateName { }`

- D) `delegate returnType delegateName(parameters) => { }`

\*\*Answer: A) `delegate returnType delegateName(parameters);`\*\*

3. \*\*What keyword is used to invoke a delegate?\*\*

- A) `call`

- B) `invoke`

- C) `execute`

- D) `delegate`

\*\*Answer: B) `invoke`\*\*

4. \*\*Which method is called when an event is triggered in C#?\*\*

- A) The event's handler method

- B) The event's delegate

- C) The event's constructor

- D) The event's delegate method

\*\*Answer: A) The event's handler method\*\*

5. \*\*What is the default access modifier for a delegate?\*\*

- A) `public`

- B) `private`

- C) `protected`

- D) `internal`

\*\*Answer: B) `private`\*\*

6. \*\*How can you add an event handler to an event in C#?\*\*

- A) Using the `+=` operator

- B) Using the `-=` operator

- C) Using the `=` operator

- D) Using the `:` operator

\*\*Answer: A) Using the `+=` operator\*\*

7. \*\*Which of the following is the correct way to declare an event in C#?\*\*

- A) `event delegateType eventName;`

- B) `event delegateType { eventName; }`

- C) `delegateType eventName { get; set; }`

- D) `delegateType eventName() { }`

\*\*Answer: A) `event delegateType eventName;`\*\*

8. \*\*What is the purpose of the `event` keyword in C#?\*\*

- A) To create a delegate instance

- B) To define an event and encapsulate the delegate

- C) To initialize a delegate

- D) To call a method asynchronously

\*\*Answer: B) To define an event and encapsulate the delegate\*\*

9. \*\*Can an event be raised outside its class in C#?\*\*

- A) Yes, always

- B) No, only within the class where it is declared

- C) Yes, but only by derived classes

- D) Yes, but only if it's public

\*\*Answer: B) No, only within the class where it is declared\*\*

10. \*\*Which delegate type is commonly used for event handling in C#?\*\*

- A) `Action`

- B) `Func`

- C) `EventHandler`

- D) `Predicate`

\*\*Answer: C) `EventHandler`\*\*

11. \*\*What is the purpose of the `Action` delegate in C#?\*\*

- A) To return a value

- B) To perform an action without returning a value

- C) To handle events

- D) To manage asynchronous operations

\*\*Answer: B) To perform an action without returning a value\*\*

12. \*\*How can you remove an event handler from an event in C#?\*\*

- A) Using the `-=` operator

- B) Using the `+=` operator

- C) Using the `=` operator

- D) Using the `!` operator

\*\*Answer: A) Using the `-=` operator\*\*

13. \*\*What is the main difference between a delegate and an event in C#?\*\*

- A) A delegate can be directly invoked, while an event cannot be directly invoked

- B) An event can be invoked, but a delegate cannot

- C) Events are more flexible than delegates

- D) There is no difference between them

\*\*Answer: A) A delegate can be directly invoked, while an event cannot be directly invoked\*\*

14. \*\*How can you declare a delegate that accepts two `int` parameters and returns an `int`?\*\*

- A) `delegate int MyDelegate(int, int);`

- B) `delegate int MyDelegate(int x, int y);`

- C) `delegate int MyDelegate(int x, int y) { }`

- D) `delegate int MyDelegate(int x, int y) => 0;`

\*\*Answer: A) `delegate int MyDelegate(int, int);`\*\*

15. \*\*Which of the following is an example of a multicast delegate?\*\*

- A) A delegate that can point to multiple methods

- B) A delegate that can only point to one method

- C) A delegate with no return type

- D) A delegate that can handle events

\*\*Answer: A) A delegate that can point to multiple methods\*\*

16. \*\*What will happen if you try to invoke an event with no subscribers?\*\*

- A) An exception will be thrown

- B) Nothing will happen

- C) The event will be subscribed automatically

- D) The program will crash

\*\*Answer: B) Nothing will happen\*\*

17. \*\*Which of the following is true about anonymous methods in C#?\*\*

- A) They can be used with events

- B) They must be named

- C) They cannot be used with delegates

- D) They can only be used with Action delegates

\*\*Answer: A) They can be used with events\*\*

18. \*\*What keyword is used to define an anonymous method in C#?\*\*

- A) `delegate`

- B) `func`

- C) `lambda`

- D) `action`

\*\*Answer: A) `delegate`\*\*

19. \*\*What is the purpose of the `EventHandler<T>` delegate in C#?\*\*

- A) To handle exceptions

- B) To provide a standard way to handle events with a parameter of type `T`

- C) To define asynchronous methods

- D) To create anonymous methods

\*\*Answer: B) To provide a standard way to handle events with a parameter of type `T`\*\*

20. \*\*In what scenario would you use a `Predicate<T>` delegate?\*\*

- A) When you need to perform a check and return a boolean value

- B) When you need to execute an action

- C) When you need to return a value from a method

- D) When you need to handle an event

\*\*Answer: A) When you need to perform a check and return a boolean value\*\*