Quiz 1 Advanced Language Features

1. What is the `auto` specifier?
   a) It is similar to `typedef` to make code more readable
   b) It is used to declare variables and functions instead of fixed types
   c) It is used to declare heterogeneous data types
   d) It is used to specialise template parameters

2. Which of the following statements concerning the `auto` specifier is true?
   a) The type of variable being declared is automatically deduced from its initializer
   b) For functions, the return type is deduced from the return statements
   c) The keyword `auto` may be accompanied by modifiers such as `const` and `&`
   d) Mixing `auto` variables and functions in one declarations is allowed

3. What is the `noexcept` specifier?
   a) It performs a compile-time check that returns true if an expression is declared to not throw any exception
   b) It specifies whether a function will throw exceptions
   c) It is a way to suppress exceptions being thrown to clients
   d) It ensures that all exceptions will be thrown from a function

4. Which of the following statements regarding the `noexcept` specifier are true?
   a) If used, it guarantees that client functions will not throw exceptions
   b) The C++98 exception specification is still supported in C++11 but it is deprecated
   c) The stack is unwound when using the C++11 exception specification
   d) It is part of a function's specification

5. What is the `noexcept` operator?
   a) It performs a compile-time check that returns true if an expression is declared to not throw any exception
   b) It serves the same objectives as the `noexcept` specifier
   c) It is not supported in C++11
   d) It specifies whether a function will throw exceptions

6. What is the `constexpr` specifier?
   a) It defines an expression that can be evaluated at compile time
   b) It specifies that the value of a variable can appear in constant expressions
   c) It specifies that the value of a variable or function can appear in constant expressions
   d) It has the same functionality as `const`

7. Which of the following statements regarding the `constexpr` specifier are true?
   a) All `constexpr` objects are `const` but not all `const` objects are `constexpr`
   b) A `constexpr` variable must be immediately constructed and assigned a value
   c) `constexpr` functions may be virtual
   d) `constexpr` values are known at compile-time while `const` values can be defined at run-time

8. What is the *alias template* (*type alias*) declaration in C++11?
   a) It is a way to create an alias that can be used anywhere instead of a (possibly complex) non-template type name
   b) It is a derived class of a template class
   c) It is a name that refers to a previously defined type (it refers to a family of types)
   d) It is used to improve code performance

9. Which of the following statements regarding the *alias template* (*type alias*) declaration are true?
   a) It can be partially and explicitly specialised
   b) It does not introduce a new type
   c) An alias template is defined using the keyword `alias`
   d) It helps in making template code more readable (for example, C++ *smart pointers*)

10. Which of the following statements regarding the *alias template* (*type alias*) declaration are true?
    a) An alias template cannot be defined in terms of another alias template
    b) An alias can be used instead of the C++98 `typedef` declaration for both template and non-template synonyms
    c) The `using` keyword simplifies the readability of function pointer declarations
    d) Aliased template and non-template classes can be members of classes (*Composition*)

11. What is a *higher-order function* in functional programming (give two answers)?
    a) It is a composition of two other functions.
    b) It takes one or more functions as arguments.
    c) It returns a function as its result.
    d) All functions in C++11 are higher-order functions.

12. Which two of the following descriptions best describe a *closure* in computer science?
    a) It is a function together with a referencing environment for the function's non-local variables.
    b) It is an anonymous function object that is generated by the C++ compiler based on a lambda function.
    c) It is a lambda function together with non-local global variables.
    d) In C++, it is a lambda function whose non-local variables are copied into the function when called.

13. Which statement best describes *currying* in computer science?
    a) It refers to the process of fixing a number of arguments to a function producing another function of small arity.
    b) It transforms a function that takes multiple arguments (or an n-tuple of arguments) in such a way that it can be called as a chain of functions each with a single argument.
    c) It takes a function with a given arity and it produces a function with a higher arity.
    d) It is a process of chaining a given function with itself a number of times.

14. Which statement best describes *partial function application* in computer science?
    a) It refers to the process of fixing a number of arguments to a function producing another function of small arity.
    b) It transforms a function that takes multiple arguments (or an n-tuple of arguments) in such a way that it can be called as a chain of functions each with a single argument.
    c) It is supported in `std::bind`.
    d) It can be implemented using lambda functions in C++.