1. A structure is a data type that can be used to group items of possibly different types into a single type
2. The dot operator is used to access elements in a structure
3. A structure tag is an identifier used after the keyword struct when defining a struct
4. The tag gives the name of the structure type and can be used in subsequemt references to the structure type. Variable of that structure type will hold the entire sequence defined by that type
5. -> are pointers dereference, it functions the same as the dot operator regarding structures and is used to access a member of a structure which is referenced by a pointer
6. It is possible to declare a structure and a pointer, so it will look something like struct test{ code} then you would create a pointer structure like: struct book \*points
7. The only operation that can be applied to struct variables is assignment, and you cannot do a equality check.
8. It is simply a structure within a structure. A structure can be declared inside another structure
9. An array if structures can be defined in collection of multipl structure variables where each variable contains information about different entities. Used to store information about multiple entities of different data types
10. Yes, in nested structures it is possible. Essentially you are just creating something to an array of an array
11. Yes, declaring an array of structure is same as declaring an array of fundamental types
12. ???
13. Yes, it allows you to create custom data types or an alias name for another data type
14. Typedef allows us to create an alias or new name for an existing type
15. Typedef is a compiler token while #define is a preprocessor token. Typedef is ended with a semicolon while #define does not. Typedef is used to give a symbolic name to the type while #define is used to create an alias of any type