1. Address operator is denoted by ampersand &,which returns the address of a variable. In example, after declaration of a pointer variable we need to initialize the pointer with a valid memory address.
2. Indirection operator is the \* and is used on a pointer variable and returns a value for the object pointed to by the variable, object is accessed indirectly through pointer
3. Pointers are a variable that stores/points the address of another variable. It is used to allocate memory dynamically
4. It is a variable containing the address of another variable, the address can be put in anpther variable in a c. Because its value points to where the value of the original variable is rather than being a value itself
5. By assigning it, in example. If you have a int x=10;, to assign say a pointer int \*p; you would use p=&x;, this is to assign the address of x to the pointer p
6. By using the \* operator u tell the compiler that the identifier ptr should declare as a pointer and the data type u specify before it tells the compiler that pointer ptr will store memory address of integer type variable
7. Function with no argument and no return value: when a function has no arguments,it does not receive any data from the calling function. When a function has arguments, it receives any data from the calling function but it returns no values
8. ???
9. The call by function copies the address of an argument into the formal parameter.inside the function the address is used to access actual argument used in call. This means that changes made to the parameter will affect the passed argument
10. Call by reference?
11. ???(call by value would normally return only 1 result?)
12. NULL pointer is a pointer that does not point to any object or function
13. It is denoted by \*\* instead of one, and it is essentially a pointer that will point to another pointer.
14. In example int \*p; <- pointer declaration then, \*p=4; this passes a value into the memory location of pointer p.
15. In call by value, copy of the actual arguments is passed to formal arguments while call by reference the location/address of the actual arguments will also reflect in actual arguments