Day 1---🡪

* Source code vs Object code
* Source code -> Given input code
* Object code -> compiled intermediate code
* Using compiler with flags
* Used for easy debugging of our code where we can move to the next/previous debug point and can see whats happening at that debug point.
* Keywords, variables
* Keywords or reserved words are used for performing predefined actions which cannot be used as variable names. Example: int, char, void.
* Variable is the name allocated to the memory location where we store data.Example int a=10; Here ‘a’ is a variable name, the value 10 is stored at some memory which can be accesse using ‘a’
* Variable scope, lifetime, initial value, modifiers
* Variable scope is the scope at which the variable can be accessed.
* Lifetime is the time during which we can access the variable(While executing the program. After completion of execution of program the variables will be garbage collected which means life time ends and when we reexecute that program again life time will be created for that variable)
* Programming Constructs(Selection construct and loop constructs)
* Programming constructs defines the order/ flow in which program statements need to be executed

1. Selection construct: Used for selection among alternatives. Example if-else
2. Loop construct: Used for executing block of statements to be executed until certain condition is met.

* Declare Array
* Static declaration: where size of the array need to be known before declaration.
* Store data in array and process elements
* Arrays store data in sequential order where each element is stored at particular index and accessing that element can be done using that index.
* We can process that elements by accessing the value at index(We need to know the index before proceesing the array element).

Day2 🡪

* Pointers, declare and initialize pointers
* We don’t use pointers in java because of insecurity of address
* In C, we declare pointer using \* for declaring and & for accessing the address. We initialialize the pointer by assigning the memory location to that pointer.
* Function definitions/invacations
* When we want to perform certain block of statements again and again then we can declare that block inside function and can access that block of code by using the name of that function.
* Call functions by passing values and address
* Call by value: Here the value of the actual arguments and those values will get stored to the formal arguments where the operations which we perform will be on formal arguments which are at different address which will not get reflected to the actual arguments.
* Call by address: Here the reference will be passed as the arguments and we perform the operations on that address which get reflected entirely.
* Dynamic memory allocation/ deallocation
* Dynamic memory allocation: It is done when we don’t know the size of memory need to be allocated. When we want to allocate memory at runtime.
* Dynamic memory deallocation: Releasing the allocated memory dynamically will sometimes free up the allocated space which can be used by other programs but in some cases we can’t get back the deallocated space(leads to loss of data). But in case of Java deallocation is done automatically by Garbage collector.
* Memory leakage
* It is a situation where a program allocates memory and losses access to that memory but that memory will remain stays as allocated which leads to wastage of memory.
* User Defined Structures
* These are the structures which are defined by the user where user can create his own structure based upon the type of data he wants to use. Example class: where we can create class with required attributes and properties, and we can access that class structure using class name.
* In C we have Structures which can be declared using struct keyword which can have heterogenious type of data.
* Access data members
* We can access the data members based upon the type of data member we declare. For static data member, it is initialized only once and can be accessed wihtout creating an object, Example main method inside class. Instance data members can be accessed by using the object of the class.
* Dynamic memory allocation for user defined structures
* C uses malloc(), calloc(), realloc() for dynamic memory allocation. We can allocate memory to the user defined structures dynamically at runtime.
* Array of pointers
* The array which consists of variables which are of type pointer, where each array variable can store the address of another variable.
* Iterate on array pointers
* Each variable of array pointers stores the address and in order to access those values of these pointer we have to deference those array pointers.

Day 3 🡪

* Implement Data structures
* Data stuctures are used for storing and organising data in an efficient manner, like linear data structure where data is stored and organised linearly.
* Linked list
* Linked list uses the concept of nodes and pointers where each node in the linked list consists of data as well as pointer to the next node. And we can access the data by using address of the node which can we obtained by traversing sequentially.
* Stack
* Stack uses Last In First Out(LIFO) where the element inserted last will be accessed first and vice versa. It uses “top” for pointing to the top most element, and “Push”, “Pop” for inserting and deleting the elements from the stack.
* Queue
* Queue uses First In First Out(FIFO) concept where the element which is inserted first will be removed first and vice versa. It uses “rear” and “front” ends for pointing towards first element and newly inserted element. It performs insertion and deletion using “enqueue” and “dequeue”.
* Binary tree
* Binary tree is hierarchial structure and accessing of elements can be done in sequential manner. It consists of root node and left, right child nodes(as the name implies each node can only contain two children- left child and right child). For inserting any element we need to know the root and if the root consists of 2 nodes then it will traverse either left or right to insert that node.